
GRANT AGREEMENT N°223866

| | |
|---|---|
| Deliverable | D02.04 |
| Nature | Report |
| Dissemination | Internal |

# D02.04 - COMMUNICATION NETWORK DESIGN (rel.2)

| | |
|---|---|
| Report Preparation Date | 24/08/2011<br>Project month: 36 |
| Authors | Federica Garin, Damiano Varagnolo, Sandro Zampieri |
| Report Version | Vn 1 |
| Doc ID Code | UNIPDP05_D02.04_28Aug11_v1 |
| Contract Start Date | 01/SEP/2008 |
| Duration | 41 months |
| Project Coordinator : | Carlos CANUDAS DE WIT, INRIA, France |


**Theme 3:**

**Information and Communication Technologies**

## SUMMARY

In this report, divided into 4 chapters, we report the final advances on WP2 and in particular on the task 3.2, "Communication network design". It corresponds to an extended version of Deliverable 2.1 since it includes some novel contributes, namely Section 2.3.

Chapter one introduces the developed arguments and presents parts of the literature review.

Chapter two is devoted to network topology design, and offers recent studies on the influence of network topologies on the performance of distributed systems, i.e. systems constituted by many interacting units. Concerning this topic, the paper [9] analyses the performance of the consensus algorithm, which is largely proposed as an efficient and low complexity tool for distributed control, estimation and optimization. The optimal topology for this algorithm, yielding fastest convergence time, is proposed. This topology is described by the de Bruijn graphs. In [27, 15] the properties of the Cayley graphs (again in relation with the consensus algorithm) are analyzed. These graphs are often used as a simple paradigm of geometric graphs, namely graphs in which the nodes are deployed in a geometric space. For this kind of graph topologies quadratic type performance indexes are often considered. Those performance indexes come into the picture when applying consensus algorithms to distributed estimation or control and yield completely different evaluation of the possible choices of the network topologies. In [35, 34] we extend the result for Cayley graphs taking into consideration a special class of geometric graphs, characterized by four purely geometric parameters. The extension is done for the particular class of reversible Markov chains due to the strong analogy they show with resistive electrical networks.

Chapter three deals with control applications, and more generally with real-time applications running over wireless sensor networks (WSNs). These systems require the design of dedicated routing protocols and control strategies to cope with the potential random delays and packet losses due to the wireless nature of the WSNs. In this chapter we then address this problem from two points of view. In the first we focus on the design of special routing strategies, namely Unicast Path Diversity (UDP) and Directed Staged Flooding (DSF), that are specifically designed for real-time application: in fact they can trade off lower end-to-end delays with higher packet loss. These two strategies, however, cannot completely remove delays randomness or packet losses. Therefore, for linear dynamical systems, we designed opportune time-varying Kalman filters that compensate both random delays and packet losses. We also address the problem of designing sub-optimal filtering strategies based on the known statistics of the packet arrival process which are computationally efficient and have limited performance degradation as compared with the optimal strategy. We also perform extensive simulations to test and evaluate both the novel routing strategies and the control/estimation strategies.

Chapter four finally reports the research activity related to the network coding. It starts treating the problem of data quantization [14, 4], the first issue to be considered when studying distributed algorithm in which the agents need to communicate through digital channels that are initially considered noiseless. In [14] the case of simple uniform quantizer is analyzed, while the more efficient quantizer, called zooming quantizer, is considered in [4]. The case of noisy digital channels is finally treated in [10] in which two possible error correcting coding methods are compared for their employment in the consensus algorithm, the first being more efficient in the exploitation of the communication resource, but more computationally demanding, while the second being less information efficient but much simpler from the computational point of view. This last paper gives, in the specific case of the consensus algorithm,

a nice view of how to treat complexity, communication and computation resources in a unified way for the solution of distributed estimation problems.

# Contents

# 1 Introduction

The performance of a networked system may depend on the characteristics of the units (sensors, actuators, computers, etc) and on the network topology describing the interactions between those units, interaction that can be of different nature, namely they can be due to physical coupling or due to communication. One part of the research effort within WP2 is about unveiling the network topology influence to networked systems behavior. In fact there are different aspects of the communication network design which will influence the behavior of the global networked control system. As described in the project description we can distinguish:

- Network topology design;
- Network management design;
- Network coding design.

In the **network topology design** it is important to understand how the communication network topology will influence the performance of the networked control system, since different topologies will ensure different information diffusion rate over the network units. In the **network management design** it is instead important to understand how the communication protocols influence the performance of the networked control system. Finally in the **network coding design** it is important to understand which impact the characteristics of the channels composing the communication network have on the performance of the networked control system. In the present report we will present a list of results on these three research activities.

As far as network topology is concerned, [9] gives an interesting contribution by showing that the optimal topology of communication the the average consensus algorithm is given by a de Bruijn graph. Indeed, in this case the standard convergence performance index, i.e. the essential spectral radius of the transition matrix, is zero and the consensus is reached in finitely many steps when the number of agents is an exact power of the out-degree of the communication graph. In [27] it has been analyzed how the network topology influences the performance of a consensus based distributed estimation algorithm. Indeed, when consensus algorithms are used in very large networks, spreading information across the whole graph requires a long time. Hence, traditional convergence analysis, studying the essential spectral radius of the transition matrix, predicts very poor performance. However, in estimation problems, it is clear that a growing number of measurements improves the quality of the estimate, and it is natural to expect such behavior even though the best estimate is approximated using distributed algorithms. Then, it is important to define a suitable performance metric, depending on the actual estimation or control problem in which the consensus algorithm is used. This allows to study how performance scales when both computation time and number of agents grow to infinity, for different communication graphs and choices of the algorithm. In [15] instead, the influence of the network topology on a more general class of quadratic indexes is analyzed. These indexes, which describe the consensus convergence evaluating the 2-norm of the consensus error and also the effect of an additive noisy on these algorithms, depend on all the eigenvalues of the transition matrix and so the scaling law with respect to the number of agents can be different from what happens to the essential spectral radius. Analytical results on asymptotic behavior of these indexes have been proposed when the communication network belongs to some families of Cayley graphs. This framework includes grids on toruses in arbitrary dimensions, which are conjectured to be a good approximation of random geometric graphs; indeed, we show simulation results supporting

this conjecture. The same conjecture is proved in [35, 34] for a different class of geometric graphs, characterized by deterministic geometric parameters.

As far as network management is concerned, in [45] it is analyzed the performance of two different routing protocols specifically designed for Wireless Sensor Networks (WSNs) for real-time estimation, control, and monitoring. These protocols are designed to compensate for the lossy nature of the wireless links and the delay from sending messages over multiple hops from the sensors to the controller. The routing protocols are designed to reduce packet delay and packet loss using either retransmissions or multicasting. For some routing topologies one protocol may be better than the other at reducing the worst case packet delay but may have a worse packet loss rate. Here, we apply mathematical tools to analytically compute the average real-time performance based on end-to-end packet delay statistics for two recently proposed routing strategies. We show that the performance is strongly related to the dynamics of the systems being estimated, and we construct a computationally efficient estimation strategy based on the delay statistics. This suggests that routing protocols are to be designed based on the specific real-time estimation and control application under consideration.

Finally, as far as network coding is concerned, [14] considers the average consensus problem on a network of digital links, and proposes a set of algorithms based on randomly switching pairwise communications and updates. The convergence properties of such algorithms is studied with the goal of answering two design questions, arising from the literature: whether the agents should encode their communication by a deterministic or a randomized quantizer, and whether they should use, and how, exact information regarding their own states in the update. In [4] instead, a new algorithm is proposed to solve the average consensus problem. The main goal of this algorithm is to obtain exact convergence despite the existence of quantized communication channels between the agents. Starting from the Zoom-in Zoom-out strategy, the equations describing the behavior of the algorithm are introduced and the asymptotic average consensus is proved. We will also show that, under a reasonable hypothesis, the algorithm parameters ensuring convergence, can be chosen regardless the number of agents. Finally, in [10] a version of the average consensus algorithm is proposed in which the agents are connected through digital noisy broadcast channels. This algorithm does not require the agents to have global knowledge of the network structure or size. Almost sure convergence to state agreement is proved, and the communication and computational complexities of the algorithms are analyzed. Both the number of transmissions and computations performed by each agent of the network are shown to grow poly-logarithmically in the desired precision. The impact of the graph topology on the algorithms' performance is analyzed as well. Moreover, it is shown how, in the presence of noiseless communication feedback, one can modify the algorithms, significantly improving their performance vs complexity tradeoff. Finally, simulations are presented confirming the theoretical results and suggesting that the presented algorithms may outperform algorithms based on decreasing gains recently proposed in the literature.

In the rest of the report we give a more detailed description of some of these contributions.

# 2 Network topology design

In this section we will propose in more details the results contained in [9] about the optimality of de Bruijn graphs for the average consensus algorithm and the results contained in [27] about the performance analysis of consensus based distributed estimation in case of Cayley graphs.

## 2.1 De Bruijn graphs for the average consensus algorithm

Assume we have a graph $\mathcal{G}$ with $N$ nodes labeled in $V = \{1, \dots, N\}$ and assume that every node $i \in V$ possesses a measurement $x_i$. The average consensus problem consists in calculating the average $x_{ave} := 1/N \sum_{i=1}^{N} x_i$ in a recursive and distributed way, allowing the nodes to communicate information along only the available edges in $\mathcal{G}$. This can be solve through the following discrete time state equations

$$x_i(t+1) = \sum_{j=1}^{N} P_{ij} x_j(t) \qquad i = 1, \dots, N\,, \tag{1}$$

where $x_i(t) \in \mathbb{R}$ is the state of the $i$-th agent at time $t$ and $P_{ij} \in \mathbb{R}$ are weighting coefficients. More compactly we can write

$$x(t+1) = Px(t), \tag{2}$$

where $x(t) \in \mathbb{R}^N$ and $P \in \mathbb{R}^{N \times N}$. The matrix $P$ is said to achieve the *average consensus* if the following conditions are satisfied:

(a) If $x(0) = \alpha \mathbf{1}$ then $x(t) = x(0)$ for every $t \in \mathbb{N}$, where $\alpha \in \mathbb{R}$ and $\mathbf{1}$ is the $N$-dimensional vector $[1, \dots, 1]^T$.

(b) Defining $x_{ave} := N^{-1}\mathbf{1^T x(0)}$, for any $x(0) \in \mathbb{R}^N$,

$$\lim_{t \to \infty} x(t) = x_{ave}\mathbf{1}.$$

Condition $(a)$ means that once consensus is reached, the agents remain at consensus. In particular observe that condition $(a)$ is equivalent to $P\mathbf{1} = \mathbf{1}$. Moreover, condition $(b)$ easily implies $\mathbf{1^T P} = \mathbf{1^T}$. It is possible to see that the average consensus problem is solved if and only if

(A) 1 is the only eigenvalue of $P$ on the unit circle centered in 0;

(B) the eigenvalue 1 has algebraic multiplicity one (namely it is a simple root of the characteristic polynomial of $P$) and $\mathbf{1}$ is its right and left eigenvector;

(C) all the other eigenvalues are strictly inside the unit disk centered in 0.

A well studied situation in the literature is when the matrix $P$ has all nonnegative entries; in that case, the conditions above exactly mean that $P$ is doubly stochastic and ergodic.
Note that the fact that in the matrix $P$ the element in position $i, j$ is different from zero means that the agent $i$ needs the state of the agent $j$ in order to update its state. This implies that we need to communicate the state $x_j(t)$ from the agent $j$ to the agent $i$.
agentThe language of graph theory is helpful to describe those communication patterns. Recall that a

directed *graph* $\mathcal{G} = (V, \mathcal{E})$ is given by set of *vertices* or *nodes* $V = (1, \dots, N)$ r and a set of *arcs* or *edges* $\mathcal{E} \subset V \times V$. The *adjacency matrix* $A$ is a $\{0, 1\}$-valued square matrix indexed by the elements in $V$ defined by letting $A_{ij} = 1$ if and only if $(i, j) \in \mathcal{E}$. Define the *in-degree* of a vertex $j$ as $\sum_i A_{ij}$ and the *out-degree* of a vertex $i$ as $\sum_j A_{ij}$.

In this context, a good description of the information flow required by a specific matrix $P$ is given by the directed graph $\mathcal{G}_P$ with set of vertices $\{1, \dots, N\}$ in which there is an arc from $i$ to $j$ whenever in the matrix $P$ the element $P_{ij} \neq 0$. In other words we adopt the convention that the presence of the edge $(i, j)$ means that $i$ observes the state $j$, agent or sends to $j$ a request for information. Note that the reverse convention is adopted in some papers. The out-degree of a node $i$ is therefore the number of nodes it observes. The graph $\mathcal{G}_P$ is said to be the *communication graph* associated with $P$. Conversely, given any directed graph $\mathcal{G}$ with set of vertices $\{1, \dots, N\}$, we say that a matrix $P$ is *compatible* with $\mathcal{G}$ if $\mathcal{G}_P$ is a subgraph of $\mathcal{G}$ (we will use the notation $\mathcal{G}_P \subseteq \mathcal{G}$). For the sake of agent clarity, it is worth noting that if a graph $\mathcal{G}$ contains the self loop $(i, i)$ it means that the $i$-th agent has access to its own state. In the sequel, we will impose the following constraint on the communication graph: the max out-degree of the nodes is $\nu$. This models the fact that communication lines are costly to establish or to operate, and every agent has the right to talk to a limited number of other agents. Note that, for compatibility with usual conventions, we consider that $\nu$ counts all arcs entering a node, including self-loops, agent, i.e., edges from a node to itself (which could be considered as 'free communication' in most technological situations). Without this constraint, the problem becomes trivial: choose agent$P = 1/N\mathbf{1}\mathbf{1}^{\mathbf{T}}$, whose underlying graph is complete, and the consensus is reached in one step. We therefore add the following constraint on $P$:

(D) Every row of $P$ contains at most $\nu$ non-zero elements.

From this point of view we would like to obtain a matrix $P$ satisfying (A),(B),(C) and (D) and minimizing a suitable performance index. The simplest performance index is the exponential rate of convergence toward the average of the initial conditions, which has been defined in [13] as follows.
Let $P$ be any matrix satisfying conditions (A),(B),(C). Then let

$$\rho(P) := \max_{\lambda \in \text{spectrum}(P) \setminus \{1\}} |\lambda|. \tag{3}$$

This quantity is called in [13] the *essential spectral radius* of $P$. As the dominant eigenvalue of $P^t$ is one and the others are smaller in magnitude than $\rho(P)^t$, the essential spectral radius says how quickly $P^t$ converges to the rank-one matrix $1/N\mathbf{1}\mathbf{1}^{\mathbf{T}}$. In this context the index $\rho(P)$ seems quite appropriate for analyzing how performance is related to the communication effort associated with a graph. The smaller the essential spectral radius, the quicker the system will converge to the average of the initial condition. We will prove that the optimal topology of communication is given by a particular family of graphs, the de Brujin's graphs [20, 49]. We will call the strategies based on such graphs shifted Kronecker strategies. We will introduce both the de Bruijn graphs and the shifted Kronecker strategy in the next sections. However, before going into them, we will provide a characterization of the best performance achievable by any matrix satisfying the conditions $(A), (B), (C)$ and $(D)$.

Let $P \in \mathbb{R}^{N \times N}$ be a matrix satisfying the conditions $(A), (B), (C), (D)$ introduced above. Observe that if $\rho(P) = 0$, then, given any initial condition $x(0)$, the average consensus is reached in a finite number of iterations of the system (2). In this case we can say that the matrix $P$ represents a deadbeat strategy for the average consensus problem. Instead if $\rho(P) > 0$, then the average consensus can be

reached only asymptotically and in this case the essential spectral radius represents a good measure of the speed of convergence of the consensus-algorithm, as already mentioned. However, even if $\rho(P) = 0$, it is possible to lower bound the number of steps necessary to $P$ to converge to the matrix $1/N\mathbf{1}\mathbf{1}^{\mathbf{T}}$. Indeed, loosely speaking, to reach the average of the initial conditions, every agent must have information about all other agents, but it can only know $\nu$ other positions in one step of time, $\nu^2$ in two steps of time, etc. Hence the propagation of information needs around $\log N/\log \nu$ steps to connect all agents. This is essentially saying that the diameter of the communication graph is at least $\log N/\log \nu$. This reasoning is stated in more formal terms in the following proposition and in the subsequent corollary.

**Proposition 1.** *Let $x_i(t+1) = f_i(x(t), t)$ for $i = 1, \ldots, N$ be a dynamical system where every $f_i(x, t)$ depends on at most $\nu$ components of $x$. Then, for every $i$, $x_i(t)$ depends on at most $\nu^t$ components of the initial condition $x(0)$.*

**Corollary 1.** *A consensus strategy for $N$ agents, where each agent observes the position of at most $\nu$ other agents at every step, cannot reach consensus in less than $\left\lceil \frac{\ln N}{\ln \nu} \right\rceil$ steps.*

Although this paper is mainly concerned with linear time-invariant strategies $x(t+1) = Px(t)$, Corollary 1 clearly holds for any memoryless, nonlinear, time-varying strategies $x(t+1) = f(x(t), t)$. A natural question arises now:"Is there a linear, time-invariant strategy that reaches consensus in a finite time, equal to the bound given by Corollary 1?" In the following section we will provide a positive answer to the above question.

We consider now a particular family of graphs, that was introduced by de Bruijn [20] in 1946. Given any pair of positive integers, $n$ and $k$, a $k$-dimensional de Bruijn graph on $n$ symbols is a directed graph representing overlaps between sequences of symbols. We denote it by $\mathcal{G}_{n,k}$. $\mathcal{G}_{n,k}$ has $n^k$ vertices, consisting of all the possible sequences of length $k$ of $n$ given symbols. If one vertex can be expressed by shifting all symbols by one place to the left and adding a new symbol at the end of another vertex, then the latter has an outgoing edge to the former vertex. Thus the set of edges is

$$\mathcal{E} = \{(i_{k-1} \ldots i_1 i_0, j_{k-1} \ldots j_1 j_0) \ : \ j_{k-1} = i_{k-2}, \ldots, j_2 = i_1, j_1 = i_0\}$$

See Fig. 1 for an example. Interesting properties of the de Bruijn graphs can be found in [49]. Here, we limit ourselves to observe that (i) if $k = 1$, then the de Bruijn graph is the complete graph and (ii) each vertex has exactly $n$ incoming and $n$ outgoing edges.

An alternative way to describe the de Bruijn graph is provided by labeling the vertices with the elements of the set $V = \left\{0, 1, 2, \ldots, n^k - 1\right\}$. In this case, $\mathcal{E}$ contains all the edges from any $i$ to $ni, ni+1, ni+2, \ldots, ni+n-1$ (modulo $n^k$). The equivalence with the previous description is obvious if we express every number $0, 1, 2, \ldots, n^k - 1$ in base $n$.
We denote the adjacency matrix by $A_{\mathcal{G}_{n,k}}$. In this paper we are interested in a normalized version of $A_{\mathcal{G}_{n,k}}$ that we define in the following way

$$\bar{A}_{\mathcal{G}_{n,k}} = \frac{1}{n} A_{\mathcal{G}_{n,k}}. \tag{4}$$

Observe immediately that, from the property (ii) of the de Bruijn graphs above mentioned, it follows that $\bar{A}_{\mathcal{G}_{n,k}}$ is a doubly stochastic matrix. The matrix $\bar{A}_{\mathcal{G}_{n,k}}$ possesses remarkable properties as a strategy for the average consensus problem, for $N = n^k$ agents, each communicationg with $\nu = n$ agents. The next theorem will explain why.

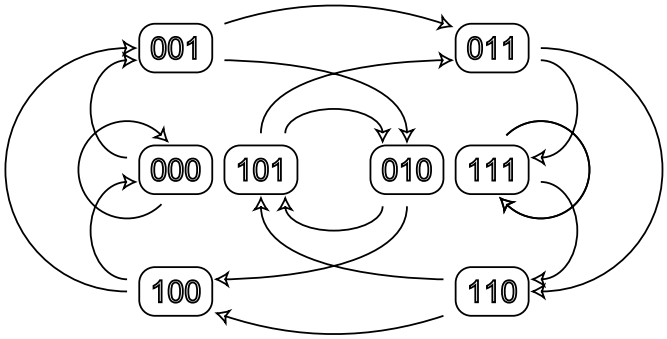


**Figure 1. The de Bruijn graph $\mathcal{G}_{2,3}$ of dimension three on two symbols. Every vertex is numbered by a number between $0$ and $7$ (here encoded in binary). The vertex $i$ is connected to $2i$ and $2i+1 \mod 8$.**

**Theorem 1.** *Let $n$ and $k$ be any two positive integers. Let $\mathcal{G}_{n,k}$ be the associated de Bruijn graph and let $\bar{A}_{\mathcal{G}_{n,k}}$ denote its normalized adjacency matrix defined as in* (4)*. Then $\bar{A}_{\mathcal{G}_{n,k}}$ satisfies the conditions $(A), (B), (C)$ and $(D)$ (for $\nu = n$). Moreover*

$$\rho\left(\bar{A}_{\mathcal{G}_{n,k}}\right) = 0 \qquad \textit{and} \qquad (\bar{A}_{\mathcal{G}_{n,k}})^k = \frac{1}{n^k}\mathbf{1}\mathbf{1}^{\mathbf{T}}.$$

*Hence, the consensus strategy represented by $\bar{A}_{\mathcal{G}_{n,k}}$ reaches consensus in $k$ steps, for any initial condition.*

Note that, the lower bound of Corollary 1 is tight for the strategy above, since $k = \frac{\ln n^k}{\ln n}$. The remarkable consequence is that among all memoryless strategies, possibly nonlinear and/or time varying, this linear time-invariant strategy is optimal in terms of speed of convergence to the average consensus. The proof of Theorem 1 is postponed to the following section, where we will introduce the shifted Kronecker strategies and we will see that $\bar{A}_{\mathcal{G}_{n,k}}$ can be viewed as a special case of them.
In the next section, we will show examples of strategies whose communication graphs are also de Bruijn graphs, but converging only asymptotically. We now show another example of an asymptotically converging strategy.

**Example 1.** *Consider the sequence of circulant matrices $P_N \in \mathbb{R}^{N\times N}$ defined by*

$$P_N = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ \frac{1}{2} & 0 & 0 & 0 & \cdots & 0 & 0 & \frac{1}{2} \end{pmatrix}. \tag{5}$$

*It is straightforward to see that in this case, see [13],*

$$\rho\left(P_N\right) = \left(\frac{1}{2} + \frac{1}{2}\cos\left(\frac{2\pi}{N}\right)\right)^{\frac{1}{2}} \simeq 1 - \frac{\pi^2}{2}\frac{1}{N^2}$$

*where the last approximation is meant for $N \to \infty$. In particular, note that $\lim_{N\to\infty} \rho(P_N) = 1$, meaning that the convergence drastically degrades as the number of nodes increases. Consider now the*

*sequence of matrices $\bar{A}_{\mathcal{G}_{2,k}}$, indexed by $k$. Observe that both for $P_N$ and $\bar{A}_{\mathcal{G}_{2,k}}$, $\nu = 2$. From Theorem 1 we have that $\rho\left(\bar{A}_{\mathcal{G}_{2,k}}\right) = 0$ for all $k$. Clearly, a larger $k$ implies that more steps are necessary to lead the state to the consensus, which, however, is always reached in finite time.*
*It is worth noting that the intrinsically slow rate of convergence of the matrices defined by* (5) *is a more general fact characterizing a broad family of matrices exhibiting symmetries: the Cayley matrices (see [13]).*

We conclude this section by underlining the fact that the de Bruijn graph has been considered in literature for efficient distribution of information in different context such as in parallel computing and peer-to-peer networks [24]. This paper can be seen as an extension of this idea to consensus problems.

### 2.2 Cayley graphs and consensus based distributed estimation

In this section, we are interested in evaluating the performance of linear time-invariant average-consensus algorithms. Typically this kind of analysis exploits results from Markov chains literature, and is focused on predicting the speed of convergence to the average, when computation time grows. There has been an extensive literature on this topic, with both analysis and optimization of asymptotic convergence speed which is given by the dominant mode of the transition matrix [13, 46]. However, we believe that when convergence to the average is not an objective per se, but is used to solve an estimation or control problem, it is important to consider different performance measures, more tightly related to the actual objective pursued.

In this paper we introduce two different functional costs which arise quite naturally in control and estimation problems. The first one is a classical LQ functional cost which evaluates the performance of the average-consensus algorithm by calculating the $L_2$ norm of a suitable variable; this cost represents a different way to evaluate the transient phase of the algorithm. In fact, also in classical control theory, there are various ways to evaluate the transient performance of a control strategy: one is based on the dominant eigenvalues, and the corresponding control methodology relies on the possibility to suitably allocate such eigenvalues; a second one is based on the $L_2$-norm of the transient and this yields to the so-called linear quadratic optimal control methodology. Our functional is this second kind of cost, for the consensus algorithm. We will show that this functional cost depends on all the eigenvalues of the transition matrix, and our main contribution will be to characterize it for some relevant families of graphs. The second cost we propose is related to the estimation error made by a network of sensors when averaging their measurements. For many families of graphs (including geometric graphs), the speed of convergence, as evaluated by the essential spectral radius, deteriorates with larger networks, but on the contrary in estimation one would expect that a broader number of measurements should improve the quality of the final estimate. We propose a natural performance measure (average error variance) and, for some families of graphs, we find its scaling laws with respect to both number of vertices and computation time, so that we can suggest useful criteria in the design of the size of large-scale sensor networks.

Before proceeding, we collect some useful definitions and we fix some notation used in this section. $\mathcal{G} = (V, E)$ denotes an *directed graph* where $V$ is the set of vertices, $N = |V|$ the number of vertices, and $E$ is the set of directed edges, i.e., a subset of $V \times V$.

A matrix $M$ is *nonnegative* if $M_{ij} \geq 0$ for all $i$ and $j$. A square matrix $M$ is *stochastic* if it is nonnegative and the sum along each row of $M$ is equal to 1. Moreover, a square matrix $M$ is *doubly-stochastic* if it is stochastic and the sum along each column of $M$ is equal to 1. Given a nonnegative

matrix $M \in \mathbb{R}^{N\times N}$, we define the induced graph $\mathcal{G}_M$ by taking $N$ nodes and putting an edge $(j,i)$ in $E$ if $M_{ij} > 0$. Given a graph $\mathcal{G}$ on $V$, the matrix $M$ is *adapted* to, or *compatible* with, $\mathcal{G}$ if $\mathcal{G}_M \subseteq \mathcal{G}$.

Now we give some notational conventions. Vectors will be denoted with bold letters. Given a vector $\boldsymbol{v} \in \mathbb{R}^N$ and a matrix $M \in \mathbb{R}^{N\times N}$, we let $\boldsymbol{v}^T$ and $M^T$ respectively denote the transpose of $\boldsymbol{v}$ and of $M$. We let $\sigma(M)$ denote the set of eigenvalues of $M$. With the symbol $\mathbf{1}$ we denote the $N$-dimensional vector having all the components equal to 1.

Given any set $A$ with finite cardinality $|A|$, $\mathbb{R}^A$ will denote the vector space isomorphic to $\mathbb{R}^{|A|}$, made of vectors where indexes are elements of $A$ instead of $\{1, 2, \ldots, |A|\}$. Analogously, $\mathbb{R}^{A\times A}$ will denote the vector space of all linear maps from $\mathbb{R}^A$ to $\mathbb{R}^A$.

### 2.2.1 Problem formulation

We start this section by briefly describing the standard discrete-time consensus algorithm. Assume that we have a set of agents $V$ and a graph $\mathcal{G}$ on $V$ describing the feasible communications among the agents. For each agent $i \in V$ we denote by $x_i(t)$ the estimate of the average of agent $i$ at $t$-th iteration. Standard consensus algorithms are built by choosing a doubly-stochastic matrix $P \in \mathbb{R}^{N\times N}$ compatible with $\mathcal{G}$ and assuming that at every step $t$ agent $i$ updates its estimates according to

$$x_i(t+1) = \sum_{j=1}^{N} P_{ij} x_j(t). \tag{6}$$

More compactly we can write

$$\boldsymbol{x}(t+1) = P\boldsymbol{x}(t), \tag{7}$$

where $\boldsymbol{x}(t)$ is the column vector whose $i$-th entry is $x_i(t)$.

It is well-known in the literature [8] that, if $P$ is primitive then the algorithm (7) solves the *average consensus problem*, namely

$$\lim_{t\to\infty} \boldsymbol{x}(t) = x_{\text{ave}}\mathbf{1}, \tag{8}$$

where $x_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$. From now on we assume the following property.

**Assumption 2.** *$P$ is a primitive doubly-stochastic matrix.* □

Traditionally the performance of the average consensus algorithm is evaluated by considering the *asymptotic convergence factor*, defined as

$$r_{\text{asym}} = \sup_{x(0)} \limsup_{t\to\infty} \left(\|x(t) - x_{\text{ave}}\|_2\right)^{\frac{1}{t}}. \tag{9}$$

It is well known [8] that, if $P$ satisfies Assumption 2 and the initial condition can be any arbitrary vector of $\mathbb{R}^N$, then $r_{asym}$ coincides with the *essential spectral radius* of $P$, that we denote by $\rho_{\text{ess}}(P)$[1].

[1]For the sake of the clarity, we recall that for a primitive doubly-stochastic matrix $\rho_{\text{ess}}(P)$ is given by the second largest eigenvalues' modulus, i.e.,

$$\rho_{\text{ess}}(P) := \max_{\lambda\in\sigma(P)\setminus\{1\}} |\lambda|. \tag{10}$$

Typically, for many graph families, when considering sequences of matrices $\{P_N\}$ of increasing size, it turns out that

$$\lim_{N\to\infty} \rho_{\text{ess}}(P_N) = 1. \tag{11}$$

Notice that equation (11) predicts a performance which degrades as the number of agents increases. This is not surprising, since intuitively one should expect that the larger is the graph, the longer is the time required to spread the information across the nodes. A mathematical characterization of (3) has been carried out for graphs exhibiting Cayley symmetries in [13] and for the random geometric graphs in [7]. In this paper we evaluate the performance of the average consensus algorithm according to two different functional costs. The first one is a classical LQ functional cost which accounts for the speed of the average consensus algorithm by calculating the $L_2$-norm of a suitable trajectory; the second one is related to the estimation error made by a network of sensors when averaging their measurements. We proceed now by presenting them separately.

### 2.2.2 Transient performance evaluation by $L_2$-norm: a LQ cost

In this subsection we assume that the initial condition $\boldsymbol{x}(0)$ satisfies the following condition.

**Assumption 3.** *The initial condition $\boldsymbol{x}(0)$ is a random variable such that $\mathbb{E}[\boldsymbol{x}(0)] = 0$ and*

$$\mathbb{E}\left[\boldsymbol{x}(0)\boldsymbol{x}^T(0)\right] = \sigma_0^2 I$$

*for some $\sigma_0^2 > 0$.* □

Without loss of generality, we will consider $\sigma_0^2 = 1$ throughout this paper.

When dealing with the average consensus problem it is convenient to introduce the following random variable

$$\Delta(t) = \boldsymbol{x}(t) - \boldsymbol{x}_{\text{ave}}\mathbf{1} = \left(\mathbf{I} - \tfrac{\mathbf{1}}{\mathbf{N}}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right)\boldsymbol{x}(\mathbf{t}),$$

where the last equality follows from the fact that, since $P$ is doubly-stochastic then $\mathbf{1}^{\mathbf{T}}\boldsymbol{x}(\mathbf{t}) = \mathbf{1}^{\mathbf{T}}\boldsymbol{x}(\mathbf{0})$ for all $t \geq 0$. Observe that $\Delta(t)$ represents the distance of $\boldsymbol{x}(t)$ from the average of the initial conditions. It is easy to see that $\Delta$ satisfies the same recursive equation as $\boldsymbol{x}$, i.e.,

$$\Delta(t+1) = P\Delta(t) \tag{12}$$

and that $\boldsymbol{x}(t) \to \boldsymbol{x}_{\text{ave}}\mathbf{1}$ if and only if $\Delta(t) \to \mathbf{0}$.

In control theory a classical way of evaluating the performance of system (12) would be considering a linear quadratic cost of the form

$$J_\Delta(P) = \frac{1}{N}\sum_{t=0}^{\infty}\mathbb{E}\left[\Delta^T(t)Q\Delta(t)\right]$$

where $Q$ is an pre-assigned semidefinite positive matrix. In our setup we assume that $Q = I$ and hence the above expression reduces to the following functional cost

$$J_\Delta(P) := \frac{1}{N}\sum_{t=0}^{\infty}\mathbb{E}\|\Delta(t)\|^2. \tag{13}$$

It is worth noting that $J_\Delta(P)$ represents also as the $L_2$-norm of the random process $\{\Delta(t)\}_{t=0}^{\infty}$.

In Section 2.2.5 we will characterize the behavior of $J_\Delta(P)$ for some relevant graph families as the number of agents $N$ varies.

We provide now a characterization of $J_\Delta$ that will be useful later on. Notice that Assumption 3 implies that $\Delta(0)$ is a random variable such that $\mathbb{E}[\Delta(0)] = \mathbf{0}$ and $\mathbb{E}\left[\Delta(0)\Delta^T(0)\right] = I - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}$. Hence,

$$\begin{aligned} J_\Delta(P) &= \tfrac{1}{N}\textstyle\sum_{t=0}^{\infty} \operatorname{trace}\left\{\mathbb{E}\left[\Delta(t)\Delta^T(t)\right]\right\} \\ &= \tfrac{1}{N}\textstyle\sum_{t=0}^{\infty} \operatorname{trace}\left[P^t\left(I - \tfrac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right)(P^t)^T\right] \end{aligned}$$

If $P$ is normal, i.e., $PP^T = P^TP$, the above expression can be written as a function of only the eigenvalues of $P$. Precisely, by observing that $P^t\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right) = \left(P\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right)\right)^t$, and that

$$\sigma\left(P\left(I - \frac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right)\right) = \{0\} \cup \sigma(P) \setminus \{1\}\,,$$

we can write

$$\begin{aligned} J_\Delta(P) &= \tfrac{1}{N}\textstyle\sum_{t=0}^{\infty}\sum_{\lambda\in\sigma(P)\setminus\{1\}} |\lambda|^{2t} \\ &= \tfrac{1}{N}\textstyle\sum_{\lambda\in\sigma(P)\setminus\{1\}} \frac{1}{1-|\lambda|^2} \end{aligned}$$

**Remark 1.** *We end this subsection by remarking that the functional cost* (13) *has been also analyzed in [47] in a different context. The authors in [47] consider a stochastic model for distributed average consensus where each node, updates its local variable with a weighted average of its neighbors' values as in* (6)*, but each new value is corrupted by an additive noise with zero mean, i.e.,*

$$x_i(t+1) = \sum_{j=1}^{N} P_{ij}x_j(t) + v_i(t), \qquad i = 1,\dots,N, \tag{14}$$

*where $v_i(t)$, $i = 1,\dots,N, t = 0,1,\dots$ are independent random variables, identically distributed, with zero mean and unit variance. To analyze the performance of the algorithm, the authors introduce the variable*

$$z(t) = \left(I - \tfrac{1}{N}\mathbf{1}\mathbf{1}^{\mathbf{T}}\right)\boldsymbol{x}(t),$$

*and the corresponding functional cost*

$$\delta_{ss}(P) := \frac{1}{N}\sum_{t=0}^{\infty}\mathbb{E}\|z(t)\|^2$$

*Observe that, due to the presence of the noise, differently from $\Delta(t)$, $e(t) \neq \boldsymbol{x}(t) - \boldsymbol{x}_{\text{ave}}\mathbf{1}$; in other words $e(t)$ quantifies the distance of the states from their current average which, in general, differs from the average of their initial conditions. Thus the mean-square deviation $\delta_{ss}(P)$ can be viewed as a measure of how well the weight matrix $P$ is able to enforce consensus (but not in general the average consensus), despite the additive errors introduced at each node at each step. Some straightforward manipulations show that, if $P$ is normal, then $\delta_{ss}(P) = J_\Delta(P)$.*

### 2.2.3 Quadratic error in distributed estimation

In this subsection we consider the following problem of distributed estimation: $N$ sensors measure the same real quantity $y$ plus independent. noises. To be more precise, if $v_i$ denotes the measurement made by the $i$-th sensor, we have that $v_i = y + n_i$ where $n_i$, $i \in \{1, \dots, N\}$, are independent zero-mean noises with the same variance $\sigma_0^2$ (without loss of generality, we will assume $\sigma_0^2 = 1$). If all the measurements were available at the same location, it is well known that the optimal estimate would be given by the mean of all measurements, i.e., $1/N \sum_{i=1}^N v_i$.

In our setup, where the sensors could be constrained by the graph $\mathcal{G}$ to communicate only with a limited number of neighbors, the average of the measurements $v_i, i \in \{1, \dots, N\}$ can be computed efficiently in a distributed way by means of an average consensus algorithm. Let $\boldsymbol{x}(0)$ be such that its $i$-th component $x_i(0)$ is equal to $v_i$. Then the estimate $x(t)$ is updated by the sensors according to (6), where $P$ is a doubly-stochastic matrix compatible with the graph $\mathcal{G}$. Clearly, under Assumption 2, $\boldsymbol{x}(t) \to \left(1/N \sum_{i=1}^N v_i\right) \mathbf{1}$. In this context, since the goal is estimating $y$, it is quite natural to introduce the error variable

$$\boldsymbol{e}(t) = \boldsymbol{x}(t) - y\mathbf{1}$$

and the corresponding quadratic functional cost

$$J_e(P, t) = \frac{1}{N} \mathbb{E}\left[\boldsymbol{e}^T(t)\boldsymbol{e}(t)\right].$$

For our problem, it is easy to show that the cost $J_e(P, t)$ can be re-written as

$$J_e(P, t) = \tfrac{1}{N} \operatorname{trace}\left((P^t)^T P^t\right)$$

If $P$ is normal, then this is equivalent to

$$J_e(P, t) = \tfrac{1}{N} \sum_{\lambda \in \sigma(P)} |\lambda|^{2t}.$$

In the next sections, we will study the asymptotic behavior of $J_e(P, t)$ when both $N$ and $t$ grow to infinity, for some families of graphs. This result is particularly relevant because it suggests the right trade-off between number of nodes and computation time in the design of large-scale sensor networks.

### 2.2.4 Families of graphs

We introduce here the main family of graphs and transition matrices we are going to consider in our analysis. First of all let's recall the definition of Cayley graphs: given a group $(G, +)$ and a set $S \subseteq G$, the Cayley graph $\mathcal{G}(G, S)$ is a directed graph with vertex set $G$ and edge set $E = \{(g, h) : h - g \in S\}$. We will consider finite graphs, with $|G| = N$, and matrices associated with such graphs, which respect the strong symmetries of the graph: we say that a matrix $P \in \mathbb{R}^{G \times G}$ (i.e. with entries labeled by indexes belonging to $G$) is Cayley if $P_{g,h} = P_{g+k,h+k} \forall g, h, k \in G$. This is equivalent to say that there exists a map $\pi : G \to \mathbb{R}$ such that $P_{h,k} = \pi(h - k)$; such function is called the generator of the Cayley matrix $P$.

Throughout this paper, we will also assume that the graph associated with $P$ is strongly connected and aperiodic, and that $P$ is stochastic, i.e., $\pi(g) \geq 0 \, \forall g \in G$ and $\sum_{g \in G} \pi(g) = 1$. Notice that a stochastic Cayley matrix is also doubly-stochastic.

In this paper, we restrict our attention to the case when $G = \mathbb{Z}_n^d$ even though most results can be generalized to any finite Abelian group. Under this assumption, the eigenvalues and eigenvectors of $P$ have the following simple expression: for any $\boldsymbol{h} = (h_1, \dots, h_d) \in \mathbb{Z}_n^d$,

$$\lambda_{\boldsymbol{h}} = \sum_{\boldsymbol{k} \in \mathbb{Z}_n^d} \pi(\boldsymbol{h}) e^{-i(\frac{2\pi}{n} h_1 k_1 + \dots + \frac{2\pi}{n} h_d k_d)}$$

is an eigenvalue with corresponding eigenvector $\boldsymbol{v}_{\boldsymbol{h}} \in \mathbb{R}^{\mathbb{Z}_n^d}$ defined by

$$\boldsymbol{v}_{\boldsymbol{h}}(\boldsymbol{k}) = \tfrac{1}{\sqrt{N}} \, e^{i(\frac{2\pi}{n} h_1 k_1 + \dots + \frac{2\pi}{n} h_d k_d)} \, .$$

Notice that such eigenvectors are orthonormal, so that $P$ is a normal matrix.

As a simple example, when $d = 1$, i.e. $G = \mathbb{Z}_N$, you obtain that $P$ is a circulant $N \times N$ matrix, with first row equal to the vector $[\pi(0), \dots, \pi(N-1)]^T$, and with eigenvalues/eigenvectors $\lambda_h = \sum_{k=0}^{N-1} \pi(k) e^{-i\frac{2\pi}{N}hk}$ and $\boldsymbol{v}_h = [1, e^{i\frac{2\pi}{N}h}, e^{2i\frac{2\pi}{N}h}, \dots, e^{(N-1)i\frac{2\pi}{N}h}]^T$, for $h = 0, \dots, N-1$.

In our analysis we want to consider families of Cayley graphs, with a growing number of vertices, but with constant degree, and with the same algebraic structure and same values for the non-zero entries of $P$. For example, we can look at a a circular graph where each node talk to itself, to its first two neighbours on the right and its first neighbour to the left, each with weight $1/3$, regardless the number of agents.

First of all, we consider a family of groups $G_n = \mathbb{Z}_n^d$, for some fixed $d$ and growing $n$; let $N = |G_n| = n^d$. Then, we have to define the neighbours and the weights. We fix a positive integer $\delta$, we define the set $D_\delta = \{-\delta, -\delta+1, \dots, +\delta\}^d$ and we fix $|D_\delta|$ real numbers $p_{\boldsymbol{h}}$, $\boldsymbol{h} = (h_1, \dots, h_d) \in D_\delta$ such that $p_{\boldsymbol{h}} \geq 0 \, \forall \boldsymbol{h}$ and $\sum_{\boldsymbol{h} \in D_\delta} p_{\boldsymbol{h}} = 1$. Then, for any $n > \delta$, we construct the Cayley matrix $P_n \in \mathbb{R}^{\mathbb{Z}_n^d \times \mathbb{Z}_n^d}$ with generator $\pi_n : \mathbb{Z}_n^d \to \mathbb{R}$ defined by $\pi_n(\boldsymbol{g}) = p_{\boldsymbol{h}}$ if there is an $\boldsymbol{h} \in D_\delta$ such that, for all $l = 1, \dots, d$ $g_l = h_l \bmod n$, and $\pi_n(\boldsymbol{g}) = 0$ otherwise. Note that for any $n \geq \delta$ $\pi_n$ is well-defined.

We can also do a similar construction taking $G = \mathbb{Z}^d$ and defining $\pi(\boldsymbol{g}) = p_{\boldsymbol{g}}$ if $\boldsymbol{g} \in D_\delta$ and $\pi(\boldsymbol{g}) = 0$ otherwise. We assume that there are enough non-zero weights $p_{\boldsymbol{h}}$ so as to ensure that the corresponding infinite graph is connected. Moreover, we assume there are self-loops, i.e., $p_{\boldsymbol{0}} \neq 0$. These two assumptions guarantee that all matrices $P_n$ of the sequence we have constructed above are primitive; also recall that $P_n$ are doubly-stochastic and normal.

We introduce here also a useful notation, defining the Laurent polynomial $p(z) \in \mathbb{R}[z_1, z_1^{-1}, \dots, z_d, z_d^{-1}]$ given by

$$p(z) = \sum_{\boldsymbol{k} \in D_\delta} p_{\boldsymbol{k}} z_1^{k_1} \dots z_d^{k_d}$$

We will refer to the above construction of a family of Cayley matrices $\{P_n\}_{n \geq \delta}$ with the short name 'Cayley matrix family associated with $p(z_1, \dots, z_d)$'. With this notation, $P_n$ has eigenvalues $\lambda_{\boldsymbol{h}} = p(e^{-i\frac{2\pi}{n}h_1}, \dots, e^{-i\frac{2\pi}{n}h_d})$, $\boldsymbol{h} = (h_1, \dots, h_d) \in \mathbb{Z}_n^d$.

Note that when we write $e^{i\frac{2\pi}{n}h_r}$ with $h_r \in \mathbb{Z}_n$, we mean that we can substitute $h_r$ with any integer which is equal to $h_r \bmod n$. Later, we will need the specific choice of $h_r \in \{0, 1, \dots, n-1\}$, which we will denote by $\boldsymbol{h} \in V_n$, $V_n = \{0, \dots, n-1\}^d$. When needed, we will actually identify vertices of the graph with $V_n$ rather than $G_n$.

The families of Cayley graphs on the group $Z_n^d$ presented above can be seen as grids on a (multi-dimensional) torus. An interesting result by Boyd et al. [6] allows to compute the eigenvalues and

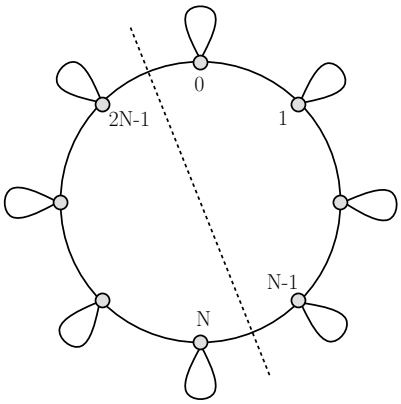


**Figure 2. Circle with $2N$ vertices and reflection axis corresponding to the map $l \mapsto 2N-1-l$, used in the construction of a line with $N$ vertices.**

eigenvectors also of grids on a cube in $\mathbb{R}^d$, which are the same as the one on a torus except that they are suitably modified at the borders.

More precisely, define the following family of matrices. Consider $P_{2n}$ a Cayley matrix on $\mathbb{Z}_{2n}^d$ associated with $p(z_1, \ldots, z_d)$, and assume that the coefficients $p_{\boldsymbol{h}}$ satisfy the following quadrantal symmetry: $p_{\boldsymbol{h}} = p_{\boldsymbol{k}}$ if $\forall i, h_i = \pm k_i$. This assumption implies that reflections $\sigma_r$ on $G_n$ defined by $\sigma_r(\boldsymbol{h}) = \boldsymbol{k}$ with $k_l = h_l$ if $l \neq r$ and $k_r = 2n - 1 - r$, are symmetries of the labeled grid on the torus. For example, Fig. 2 shows the axis of reflection of $\sigma_1$ for the case $d = 1$. It is convenient here to identify $G_n$ with the set $V_n = \{0, \ldots, n-1\}^d$, and consider $\sigma_r : V_n \to V_n$.

Now denote by $H$ the group generated by $\sigma_1, \ldots, \sigma_d$ and consider, for all $\boldsymbol{g} \in V_n \subseteq V_{2n}$, the orbit $O_{\boldsymbol{g}} = \{\eta(\boldsymbol{g}) : \eta \in H\} \subseteq V_{2n}$. Finally, define $\overline{P}_n : \mathbb{R}^{V_n} \to \mathbb{R}^{V_n}$ by $(\overline{P}_n)_{\boldsymbol{h},\boldsymbol{k}} = \sum_{\boldsymbol{l} \in O_{\boldsymbol{k}}} P_{\boldsymbol{h},\boldsymbol{l}}$, for all $\boldsymbol{h}, \boldsymbol{k} \in V_n$. Notice that $\overline{P}_n$ is symmetric and that, apart from the borders, $\overline{P}_n$ associates to edges of the grid the same coefficients that $P_n$ associates to edges of the grid on the torus.

We will refer to the above construction of a family of matrices $\{\overline{P}_n\}_{n \geq \delta}$ with the short name 'grid matrix family associated with $p(z_1, \ldots, z_d)$'.

Using [6, Prop. 3.2], we can find the eigenvalues of $\overline{P}_n$:

$$\bar{\lambda}_{\boldsymbol{h}} = p(e^{i\frac{\pi}{n}h_1}, \ldots, e^{i\frac{\pi}{n}h_d}), \qquad \boldsymbol{h} \in V_n\,.$$

The random geometric graph is a random undirected graph drawn on a bounded region, e.g., the $d$-dimensional unitary cube $[0,1]^d$. It is generated by

- placing vertices at random, uniformly and independently inside the region, and
- connecting two vertices if and only if the euclidean distance between them is at most a pre-assigned threshold $r$.

The random geometric graph was first introduced in [28] and has been deeply studied under a communications and information-theoretic point of view in [30]. It has recently witnessed a large interest in many applications; particularly it has been successfully used to model wireless communication [25]. Given a

random geometric graph $\mathcal{G}(V, E)$, it is possible to build a doubly stochastic matrix $P$ according to the *Metropolis weights* rule [48]; precisely, if $P_{ij}$ denotes the element of $P$ in the $i$-th row and in the $j$-th column we will have

$$P_{ij} = \begin{cases} \frac{1}{1+\max\{d_i,d_j\}} & \text{if } (i,j) \in E \\ 1 - \sum_{(i,k)\in E\setminus\{(i,i)\}} P_{ik} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

where $d_i = |\mathcal{N}_i \setminus \{(i,i)\}|$ with $\mathcal{N}_i = \{j \in V \,|\, (i,j) \in E\}$. In other words the weight on each edge is one over one plus the largest degree at its two incident vertices, and the self-weights are chosen so the sum of weights at each node is 1.

### 2.2.5 Main results

We state here our main theoretical results: an asymptotic analysis of the proposed quadratic indexes for the families of Cayley graphs and of grids described in previous section. The proofs can be found in the Appendix.

**Proposition 2** (LQ cost asymptotics)**.** *Given $\{P_n\}_{n\geq\delta}$ a Cayley or a grid matrix family associated with $p(z_1,\ldots,z_d)$, there exist $C_d, C'_d > 0$ (depending only on $d$) such that:*

- *if $d = 1$,*
$$C_1 N \leq J_\Delta(P_n) \leq C'_1 N\,;$$
- *if $d = 2$,*
$$C_2 \log N \leq J_\Delta(P_n) \leq C'_2 \log N\,;$$
- *if $d \geq 3$,*
$$C_d \leq J_\Delta(P_n) \leq C'_d\,.$$

□

To give a better understanding of the above Theorem, we propose an example illustrating an interestig comparison between the behavior of the functional cost $J_\Delta$ and of the essential spectral $\rho_{\text{ess}}$ as $n \to \infty$ of a particular sequence of Cayley graphs. We will see how the evaluation of the performance of the average consensus algorithms, in the asymptotic regime $n \to \infty$, is strictly related to the choice of the functional cost.

**Example 2.** *Consider the sequence of Cayley matrices $\{P_n\}$ built as follows. For each $n$, let $\mathcal{G} = \mathbb{Z}_n^3$ and let $S = \big\{(0,0,0),\ (1,0,0),\ (0,1,0),\ (0,0,1),\ (-1,0,0),\ (0,-1,0),\ (0,0,-1)\big\}$. Moreover let $\pi(g) = \frac{1}{7}$ for all $g \in S$. It is well known (see [13]) that, in this case,*

$$\rho_{\text{ess}}(P_n) \geq 1 - \frac{C}{N^3}$$

*where $C$ is a constant independent from the topology of the graphs. From the above inequality it turns out that, if we consider as functional cost the* asymptotic covergence factor *defined in* (9)*, then the performance of the average consensus algorithms associated to the sequence $\{P_n\}$ degrades drastically as $n \to \infty$. Instead, Proposition 2 guarantees the existence of constants $C_3$ and $C'_3$ such that $C_3 \leq J_\Delta(P_n) \leq C'_3$ for all $n$.* □

**Proposition 3** (Quadratic estimation error asymptotics)**.** *Given $\{P_n\}_{n\geq\delta}$ a Cayley or a grid matrix family associated with $p(z_1,\dots,z_d)$, there exist constants $c_0 = c'_0 = 1, c_1,\dots,c_d,c'_1,\dots,c'_d > 0$ and $k \in (0,1)$ such that*

$$\max_{l=0,\dots,d} \frac{1}{n^{d-l}}\frac{c_l}{t^{l/2}} \leq J_e\left(P_n,t\right) \leq k^t + \sum_{l=0,\dots,d} \frac{1}{n^{d-l}}\frac{c'_l}{t^{l/2}}$$

□

**Corollary 2.** *Given $\{P_n\}_{n\geq\delta}$ a Cayley or a grid matrix family associated with $p(z_1,\dots,z_d)$, there exists constants $n_0 \in \mathbb{N}$, $k_1, k_2 > 0$ such that, for all $n \geq n_0$,*

$$k_1 \max\left\{\frac{1}{N},\frac{1}{t^{d/2}}\right\} \leq J_e\left(P_n,t\right) \leq k_2 \max\left\{\frac{1}{N},\frac{1}{t^{d/2}}\right\}$$

□

Notice that if the average of all sensors' measurements was performed in a centralized way, the exact average thus obtained would be the best possible estimate of the measured value (under the simple model we are considering), but it would still have error variance $1/N$. Thus, it is not surprising to find a term $1/N$ in the asymptotic behavior of the decentralized algorithm running on grids. What is more interesting is to look at the dependence on $N$ and $t$, which shows that a sensible design of the number of nodes should take into account also the computational time allowed for communication and computation. In fact, Corollary 2 clearly shows that when both $t$ and $N$ grow there are two very different regimes, a first one with $N \ll t^{d/2}$, where the error decays as $1/N$, and a second regime with $N \gg t^{d/2}$ where regardless the number of nodes the cost is dominated by a term scaling as $1/t^{d/2}$. Finally, it is interesting to notice that, despite $\rho_{\text{ess}} \to 1$ for $N \to \infty$ would suggest that these families of graphs have decreasing performance for growing number of agents, indeed it is clear that a bigger number of measurements can improve the quality of the estimate, and in fact $\lim_{t,N\to\infty} J_e\left(P_n,t\right) = 0$.

Now we focus on the other relevant family of graphs we are dealing with in this paper, the random geometric graph. While several probabilistic results are known about the number of components in the graph as a function of the threshold $r$ and the number of vertices $N$ (see e.g. the monograph [38]), no comprehensive theoretical characterization has been provided yet for the behavior of the eigenvalues of doubly-stochastic matrices associated to random geometric graphs. In this direction only few results are present so far. It is worth citing them briefly. In [7], the authors first prove some regularity properties on the degrees of the nodes of random geometric graphs; then, relying on these results, they find a lower bound for the mixing time of random walks on random geometric graphs with the mixing time of random walks on regular grids on torus (the mixing time is related to the essential spectral radius of the transition matrices associated). In [39] an asymptotic spectral concentration result is presented. Families of random geometric graphs of increasing size $N$ are considered; they are built on $[0,1]^d$ with a threshold $r$ which is assumed to tend to 0 as $N \to \infty$, but in such a way that the graphs so obtained have and increasing degree and are almost surely connected. Under this assumption it is shown that the spectrums of the transition matrices of the random walks on these families of graphs converge, as $N \to \infty$, to those of the graphs on deterministic grids.

In this section we provide some numerical results characterizing the behavior of the cost $J_\Delta$ and $J_e$ for random geometric graphs. Interestingly, we will bring to light further similarities between the random geometric graphs and the deterministic grids.

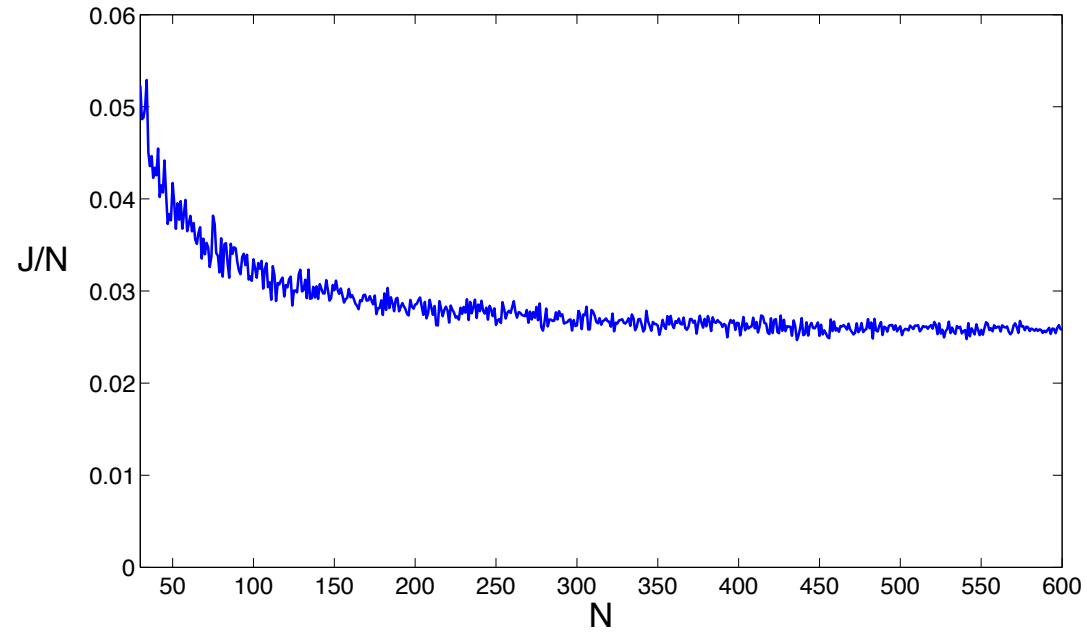


**Figure 3. Behavior of $\frac{J_\Delta}{N}$ for $d = 1$.**

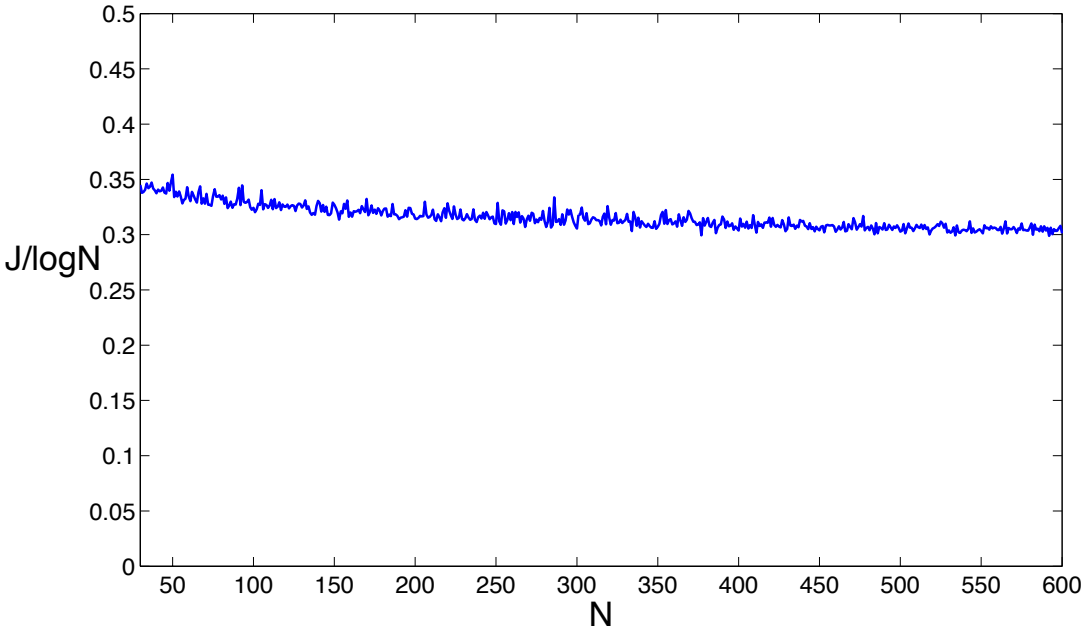


**Figure 4. Behavior of $\frac{J_\Delta}{\log N}$ for $d = 2$.**

Figures 3, 4, 5 and 6 refer to $J_\Delta$. Precisely, in Figure 3 we depicted the behavior of $J_\Delta/N$ for $d = 1$, in Figure 4 the behavior of $J_\Delta/\log N$ for $d = 2$ and in Figures 5 and 6 the behavior of $J$ for $d = 3$

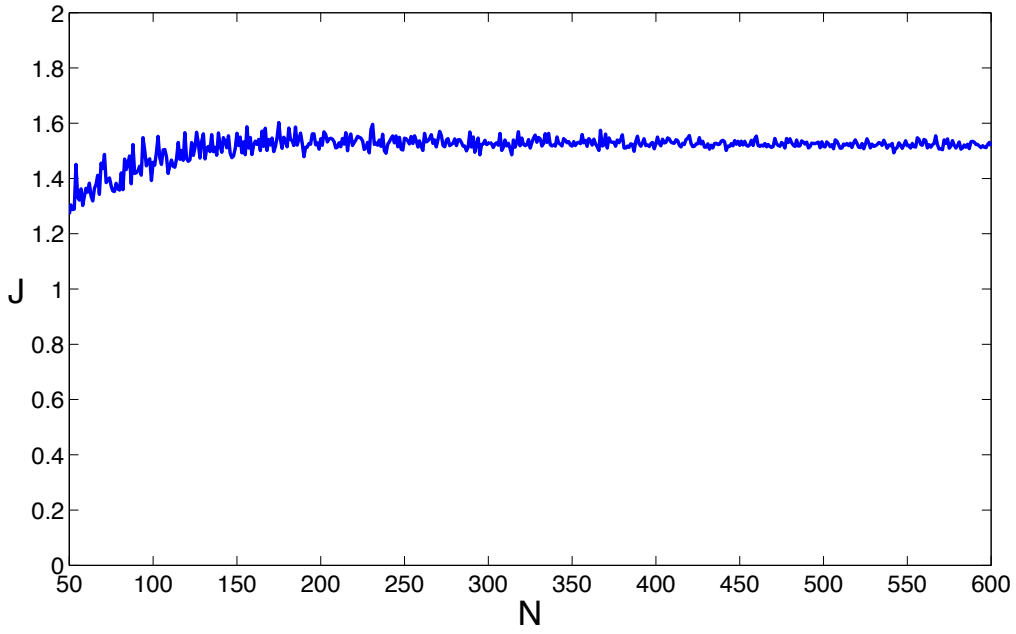


**Figure 5. Behavior of $J_\Delta$ for $d = 3$.**

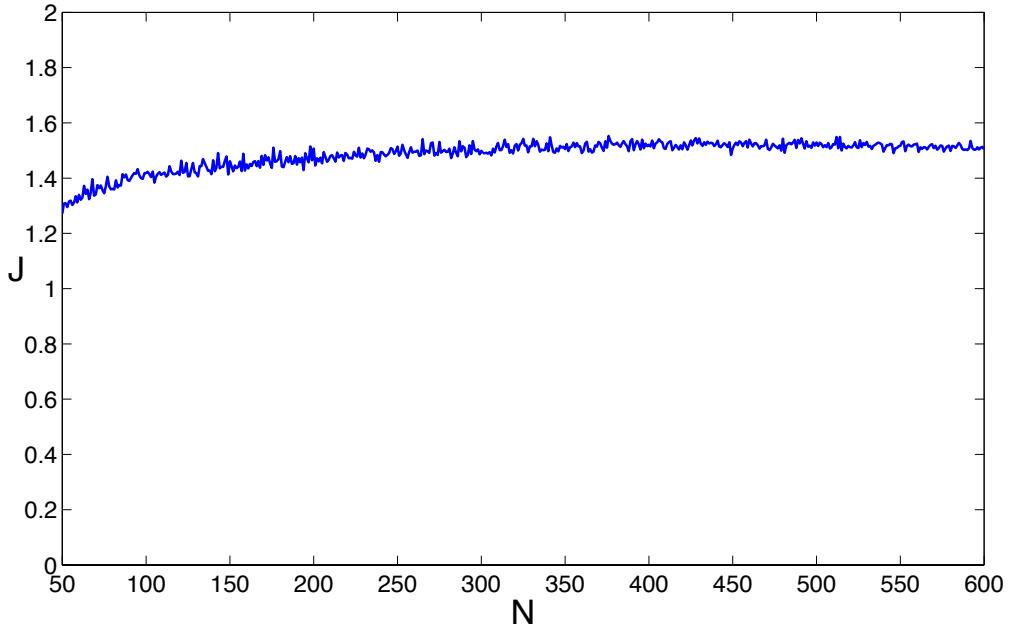


**Figure 6. Behavior of $J_\Delta$ for $d = 4$.**

and $d = 4$, respectively. For each value of $d$ we run simulations from $N = 50$ up to $600$. We consider families of connected graphs of increasing size obtained with a decreasing threshold $r$ in a such a way that the average size of the neighborhood of the nodes is kept almost constant independently from the

value of $N$ (in this specific simulations close to 12 for any value of $d$). For each value of $N$ we calculated the value of the plotted variable as the mean of 50 trials.

From Figures 3, 4, 5, 6, we can infer that $J_\Delta$ increases linearly for $d = 1$, logarithmically for $d = 2$, whereas it becomes asymptotically constant for $d = 3$ and $d = 4$. The analogy with Proposition 2 is evident.

Figures 7, 8, 9 and 10 provide numerical results for $J_e$. Again we run simulations for $N = 50$ up to 600 by considering families of connected graphs of increasing size built with a decreasing threshold $r$ as in the previous set of simulations; however, in this case all the figures refer to the 2-dimensional case $d = 2$. Our aim is to underline the different scaling when both $t$ and $N$ grow, with $t$ being different functions of $N$: $t$ constant, $t = \sqrt{N}$, $t = N$, $t = N^{3/2}$. In Figure 7 we plotted the behavior of $J_e(P_N, 20)$, in Figure 8 the behavior of $\sqrt{N}\, J_e\left(P_N, \sqrt{N}\right)$, in Figure 9 the behavior of $N\, J_e\left(P_N, N\right)$ and in Figure 10 the behavior of $N\, J_e\left(P_N, N^{3/2}\right)$. From the drawn plots, we can deduce that $J_e$ evaluated for the random geometric graphs, exhibits a behavior very similar to the one stated in Corollary 2.

Finally, these numerical results emphasize further evident similarities between the spectral behavior of the transition matrices built on random geometric graphs and the spectral behavior of the transition matrices associated to the deterministic grids.

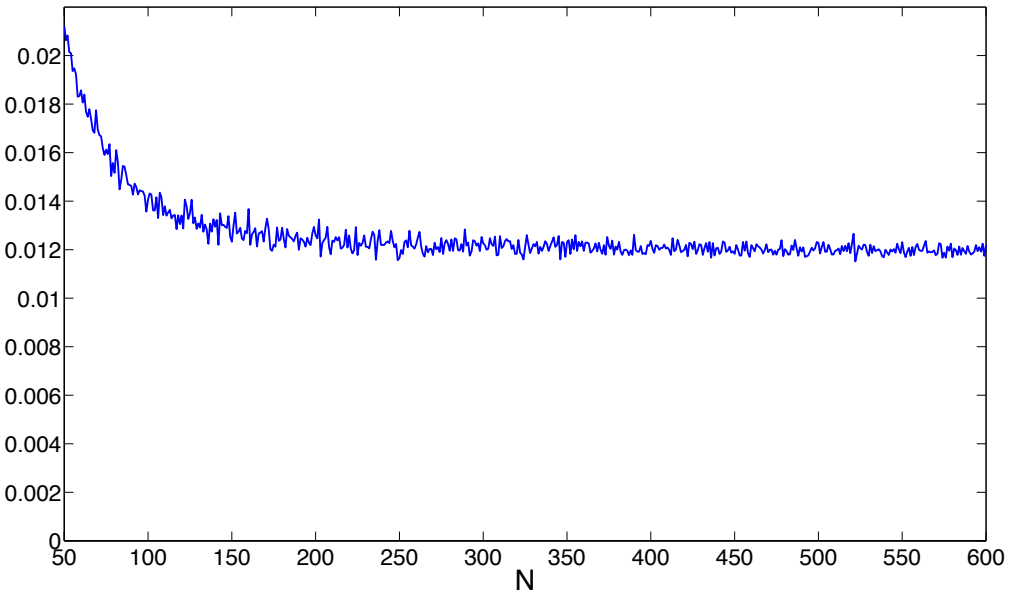


**Figure 7. Behavior of $J_e(P_N, 20)$.**

### 2.3 A resistance–based approach for the analysis of the $L_2$ cost

In this section we take into consideration the particular case of $P$ associated with a reversible Markov chain. It is known that we can interpret the row–stochastic matrix $P$ as the matrix of transition probabilities of a Markov chain. If $\pi^T$ is the left invariant normalized eigenvalue of $P$, namely $\pi^T P = \pi^T$ and

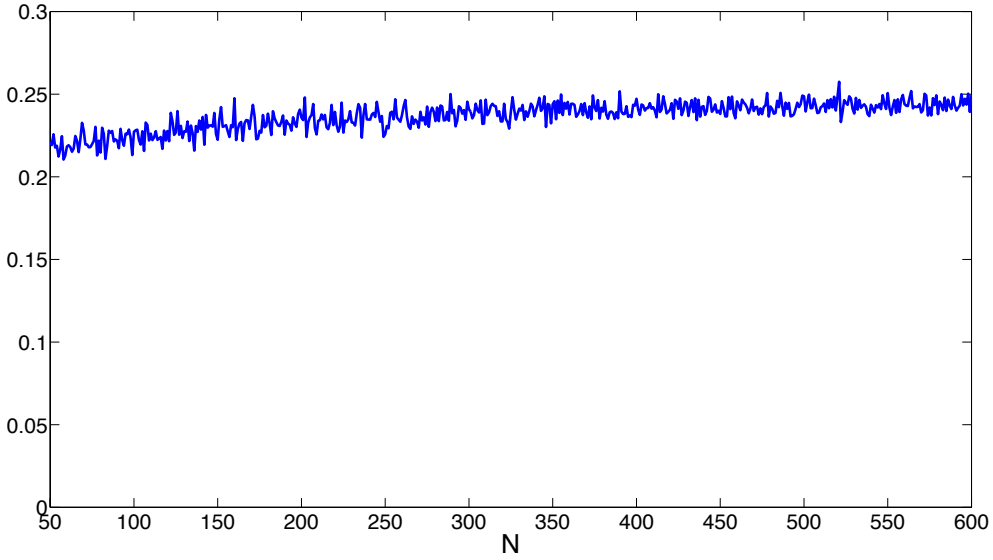


**Figure 8. Behavior of $\sqrt{N}\, J_e(P_N, \sqrt{N})$.**

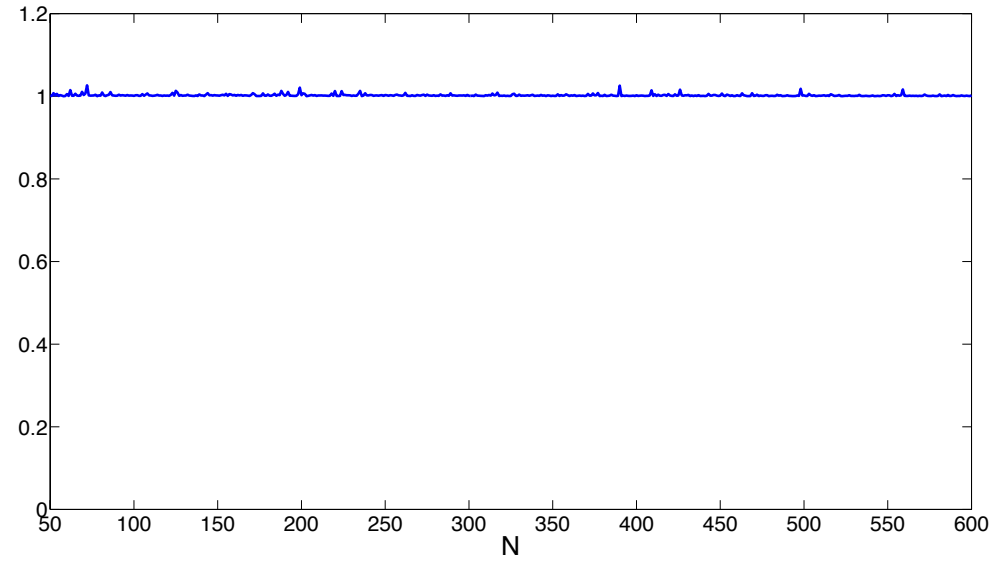


**Figure 9. Behavior of $N\, J_e(P_N, N)$.**

$\pi^T \mathbf{1} = 1$, then we say that the Markov chain is reversible if

$$\pi_u P_{uv} = \pi_v P_{vu},\ \forall\, (u,\, v) \in V \times V$$

where we recall that $V$ is the set of nodes of the graph. A matrix $P$ associated with a reversible Markov chain will be called a reversible consensus matrix. We always consider strongly connected graphs, for which it can be shown that $\pi_u > 0$ for any $u \in V$, this implying that the graph is undirected since by

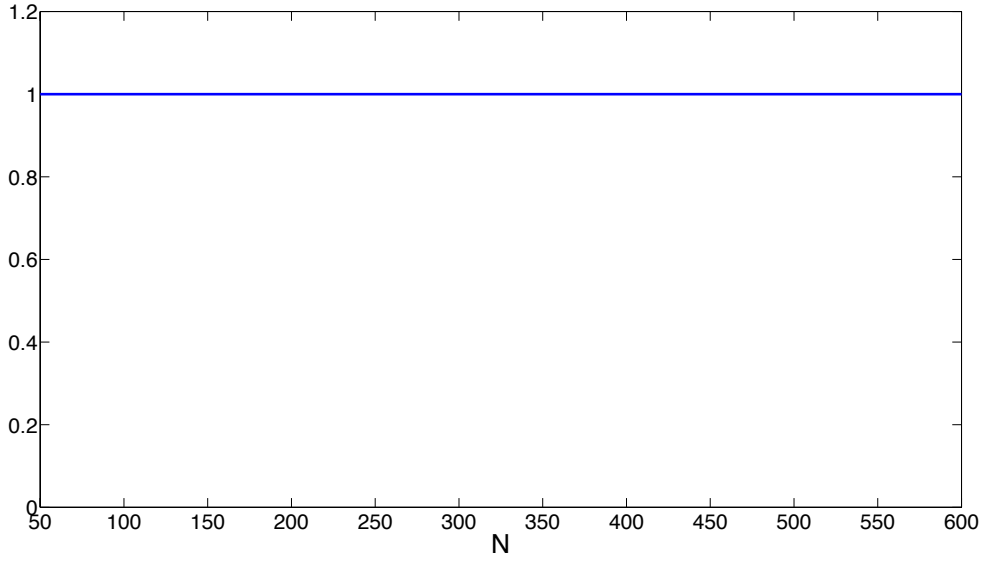


**Figure 10. Behavior of $N\, J_e(P_N, N^{3/2})$.**

the reversibility condition we have $P_{uv} > 0$ if and only if $P_{vu} > 0$, namely $(u,\, v) \in E$ of and only if $(v,\, u) \in E$.

Even if we cannot address full generality and we have to limit to the reversible case, the result is still valuable since it covers the symmetric case, thus for example $P$ could be produced using Metropolis-Hasting weighs, and uniform weights, in which all the neighbors of a node $i$, included $i$ itself, are weighted the same way.

The performance cost we study is the LQ cost introduced in Eq. 13, which we write here equivalently

$$J(P) = \frac{1}{N} \sum_{t \geq 0} \mathbb{E}\left[ \|\boldsymbol{x}(t) - \boldsymbol{x}(\infty)\|^2 \right], \tag{15}$$

where we assume that the initial condition is drown randomly according to a distribution with zero mean and unitary variance.

### 2.3.1 Electrical Analogy

In this paragraph we briefly present the analogy between Markov chains and resistive electrical networks, which dates back to the work of Doyle and Snell [21]. It allows simpler proofs for some properties of the Markov chain, and gives a strong physical intuition on its characteristics.

#### Electrical networks: definition and notations

By resistive electrical network we mean a pair $(\mathcal{G},\, c)$, or equivalently $(\mathcal{G},\, r)$, where $\mathcal{G}$ is an undirected graph with $N$ vertices, and $M$ edges, and $c$ is a function $c : E \to \mathbb{R}^+$, where $c(e)$ is called the conductance of the edge $e$. Analogously, $r$ is a function $r : E \to \mathbb{R}^+$, $r(e)$ is called the resistance of the edge $e$ and it holds true $r(e) = 1/c(e)$. Sometimes we will abuse the notation extending the

domain of $c$ to any element of $V \times V$, with the convention that nonexisting edges have zero conductance and infinite resistance. With this convention we can write $c : V \times V \to \mathbb{R}^+$ and $c_{uv}$ to mean $c((u, v))$, and notice thus that the electrical network is fully represented by the symmetric matric $C \in \mathbb{R}^{+N \times N}$ whose $(u, v)$-entry is $c_{uv}$.

We will also adopt the following convention. Even if $\mathcal{G}$ is undirected, for any edge we choose arbitrarily and w.l.o.g. one of the two directions of the edge. Then we define the incidence matrix $A \in \{0, \pm 1\}^{M \times N}$ as follows

$$A_{eu} = \begin{cases} -1 & \text{if } e = (*, u) \\ 1 & \text{if } e = (u, *) \\ 0 & \text{otherwise} \end{cases} . \tag{16}$$

As it will be clear, the direction we have chosen is only required in order to assign a sign to the flow of current in each edge.

With this notation in mind, we define the matrix $\mathcal{C} \in \mathbb{R}^{M \times M}$ as the diagonal matrix whose $(e, e)$-entry is the conductance of $e$, namely $\mathcal{C}_{ee} = c(e), \forall\, e \in E$ and $\mathcal{C}_{ee'} = 0$ if $e \neq e'$. Using this notation we obtain

$$[A^T \mathcal{C} A]_{uv} = \begin{cases} c_u & \text{if } u = v \\ -c_{uv} & \text{if } (u, v) \in \mathcal{E} \\ 0 & \text{if } (u, v) \notin \mathcal{E} \end{cases} ,$$

where $c_u = \sum_{v|(v,u)\in E} c_{uv}$ is the sum of all the conductances of the edges incoming in $u$.

### Electrical networks and Markov chains

We associate to any reversible consensus matrix $P$ with left invariant normalized eigenvector $\pi^T$ a resistive electrical network whose graph $\mathcal{G}$ is the same as the graph of $P$, and where we set the conductances as

$$c_{uv} = N \pi_u P_{uv}$$

where $N$ is the number of agents. Notice that the reversibility conditions implies $c_{uv} = c_{vu}$, which is of course needed in a resistive electrical network. If $\mathcal{C}$ and $A$ are defined as above, we can recover the matrix $P$ by simply setting

$$L = \frac{1}{N} \Pi^{-1} A^T \mathcal{C} A$$

and noticing that $P = I - L$. The matrix $L$ is the so called Laplacian of $P$.

### Electrical networks and effective resistances

Let $\boldsymbol{i} \in \mathbb{R}^N$ be given such that $\boldsymbol{i}^T \mathbf{1} = 0$, and interpret the $k$-th entry of $\boldsymbol{i}$ as the current which is injected (or extracted if negative in sign) in the $k$-th node of the network from an external source. For example, if $a$ ampere are injected in node $u$ and $a$ ampere are extracted from node $v$, we have $\boldsymbol{i} = a(\boldsymbol{e}_v - \boldsymbol{e}_u)$, where $\boldsymbol{e}_k$ denotes the element of the canonical base of $\mathbb{R}^N$ having a one in position $k$.

We denote by $\boldsymbol{j} \in \mathbb{R}^M$ and $\boldsymbol{v} \in \mathbb{R}^N$ respectively the current flow on the edges and the potentials at the node which are produced at steady state in the network by injecting $\boldsymbol{i}$. The previously defined matrices

$A$ and $\mathcal{C}$ allow us to compactly write Kirchhoff's node law and Ohm's law as the system

$$\begin{cases} A^T \boldsymbol{j} = \boldsymbol{i} \\ \mathcal{C} A \boldsymbol{v} = \boldsymbol{j} \end{cases} \tag{17}$$

where the first equation says that the total flow entering in any nodes equals the total flow exiting from it, while the second is, row by row, Ohm's law $v_u - v_v = r_{uv} j_{uv}$, $(u,\, v) \in E$.

We say that we can solve the electrical network if we find the solutions $\boldsymbol{j}$ and $\boldsymbol{v}$ of Eq. 17, which are assured to exist by Maxwell's Theorem [21]. Putting the two together and dropping $\boldsymbol{j}$ we obtain the electrical equation

$$A^T \mathcal{C} A \boldsymbol{v} = \boldsymbol{i}. \tag{18}$$

A further constraint on $\boldsymbol{v}$ is needed in order it to be uniquely determined. One can choose $\pi \in \mathbb{R}^+$ such that $\pi^T \mathbf{1} = 1$ and impose

$$\pi^T \boldsymbol{v} = 0. \tag{19}$$

We introduce now the so called Green matrix of $P$, defined as

$$G = \sum_{t \geq 0} P^t - \mathbf{1}\pi^T.$$

Notice that it depends on the matrix $P$ only. By means of such matrix, it is possible to show that

$$\boldsymbol{v} = \frac{1}{N} G \Pi^{-1} \boldsymbol{i}, \tag{20}$$

which gives a closed formula for $\boldsymbol{v}$ in terms of the injected current and of the chosen constraint for $\boldsymbol{v}$.

We conclude this section defining the effective resistance of a pair of nodes in the electrical network $(\mathcal{G},\, c)$. Given $u$ and $v$ in $V$, we say that the effective resistance among them is

$$\mathcal{R}_{uv}(\mathcal{G},\, c) = \boldsymbol{i}^T \boldsymbol{v} \tag{21}$$

once we set $\boldsymbol{i} = e_u - e_v$, namely once we inject $1$ Ampere in $u$ and extract $1$ Ampere from $v$. This is the equivalent resistance in a network in which the whole network is reduced to the edge $(u,\, v)$. By Eq. 20, it is now immediate to obtain

$$\mathcal{R}_{uv}(\mathcal{G},\, c) = \frac{1}{N} (e_u - e_v)^T G \Pi^{-1} (e_u - e_v). \tag{22}$$

The reason for which we consider reversible Markov chains and their analogy with electrical networks is that, as it will be proven further on, the performance cost we are interested in can be rewritten in terms of effective resistances in the corresponding electrical network. On the other hand, effective resistances present several monotonicity properties which are not equally clear when dealing with the matrix $P$. By making use of Rayleigh's monotonicity law and of some results proven in [5], we can in fact prove that if we have an electrical network $\mathcal{G},\, \rfloor$ in which all the resistances are such that $r_{ij} \in [r_{\min},\, r_{\max}]$, then

$$r_{\min} \mathcal{R}_{uv}(\mathcal{G},\, 1) \leq \mathcal{R}_{uv}(\mathcal{G},\, c) \leq r_{\max} \mathcal{R}_{uv}(\mathcal{G},\, 1),\, \forall\, (u,\, v) \in V \times V.$$

This justify the notation $\mathcal{R}_{uv}(\mathcal{G})$ which we use to denote the effective resistance on the graph $\mathcal{G}$, when all the resistances are set to $1$ Omh.

Another interesting result from [5] deals with $h$-fuzzes. Given an integer $g \geq 1$ and a graph $\mathcal{G}$, we call $h$-fuzz of $\mathcal{G}$, and we denote by $\mathcal{G}^{(h)}$, a graph with the same set of nodes, $V^{(h)} = V$, and with an edge among two nodes $u$ and $v$ as soon as the graphical distance $d_{\mathcal{G}}(u, v)$[2] among $u$ and $v$ in $\mathcal{G}$ has length at most $h$

$$E^{(h)} = \{(u, v) \in V \times V : d_{\mathcal{G}}(u, v) \leq h\}.$$

It is easy to see that if $\mathcal{G}$ is the graph associated to a consensus matrix $P$ in which all the diagonal entries are strictly positive, then $\mathcal{G}^{(2)}$ is the graph associated to $P^2$.

The result states that if $\mathcal{G}$ is a graph in which every node has at most $\delta$ neighbors, then

$$\frac{1}{h\delta^h}\mathcal{R}_{uv}(\mathcal{G}) \leq \mathcal{R}_{uv}(\mathcal{G}^{(h)}) \leq \mathcal{R}_{uv}(\mathcal{G})$$

where recall that $R_{uv}(\mathcal{G})$ is the effective resistance in electrical network whose graph is $\mathcal{G}$ and all the resistances are set to 1, and analogously for $R_{uv}(\mathcal{G}^{(h)})$.

In conclusion, effective resistances are characterized by a high grade of monotonicity. Moreover, we can say that if $h$ is "low", then the effective resistance in a graph and in its $h$-fuzz are essentially the same.

### 2.3.2 Main result

Let $P$ be a reversible consensus matrix with left invariant normalized eigenvector $\pi^T$, and consider the electrical network $(\mathcal{G}^{(2)}, c^P)$ in which $\mathcal{G}$ is the graph associated with $P$ and $c^P_{uv} = N\pi_u[P^2]_{uv}$. The electrical analogy implies that

$$\frac{\pi^3_{\min}N^2}{\pi_{\max}}\bar{\mathcal{R}}(\mathcal{G}^{(2)}, c^P) \leq J(P) \leq \frac{\pi^3_{\max}N^2}{\pi_{\min}}\bar{\mathcal{R}}(\mathcal{G}^{(2)}, c^P)$$

where

$$\bar{\mathcal{R}}(\mathcal{G}^{(2)}, c^P) = \frac{1}{N^2}\sum_{(u,v)\in E^{(2)}}\mathcal{R}_{uv}(\mathcal{G}^{(2)}, c^P)$$

is the average effective resistance in $(\mathcal{G}^{(2)}, c^P)$.

This result is important since it expresses the performance cost we are interested in as a function of an average effective resistance, on which we can apply the monotonicity properties we have stated above. Notice moreover that in case $P$ is symmetric, for which $\pi_u = \frac{1}{N}, \forall\, u \in V$, the previous inequality particularizes to

$$J(P) = \bar{\mathcal{R}}(\mathcal{G}^{(2)}, c^P),$$

which is the result one can find in [35].

We conclude this paragraph stating the following proposition. Its importance lies on the fact that it allows one to rewrite the cost as a function of the average effective resistance of the graph $\mathcal{G}$ only, at the mild cost of assuming that the nonzero entries of $P$ lie in some interval.

**Proposition 4.** *Let $P$ be a reversible consensus matrix with left invariant measure $\pi^T$ associated with a graph $\mathcal{G}$. Assume all the nonzero entries of $P$ lie in the interval $[p_{\min}, p_{\max}]$, and the maximum number*

[2]The graphical distance among $u$ and $v$ is defined as the length of the minimum path connecting $u$ and $v$

*of neighbors of any node be $\delta$. Let $\mathcal{G}$ represent the electrical network in which all the resistances are set to 1 Ohm. Then there exist two values $c_1$ and $c_2$ such that*

$$c_1\bar{\mathcal{R}}(\mathcal{G}) \leq J(P) \leq c_2\bar{\mathcal{R}}(\mathcal{G}) \tag{23}$$

*where $c_1$ and $c_2$ depend on $p_{\min}$, $p_{\max}$, $\delta$ and on the products $\pi_{\min}N$ and $\pi_{\max}N$.*

The importance of this proposition lies in the following argument. A very common goal when studying consensus algorithms is to understand how some specific performance cost behave as a function of the number of nodes of the graph. In other words, it is common to take a family of growing graphs all characterized by some property, and study how the dimension influences the cost. To give a simple example, it is known that the rate of convergence to consensus is the second largest eigenvalue of the matrix $P$, and that it converges to the value 1 for many classes of graphs. To understand *how fast* it converges to 1 gives some insight on the diffusion velocity of information in the graph. This is what has been done in the previous sections in the case of Cayley graphs.

With this respect, Theorem 4 is unfortunately a partial result. We can if fact find families of graphs growing in the number $N$ of nodes with associated reversible consensus matrices $P$ whose nonzero entries lying in an interval independent on $N$ and with a bounded (in $N$) neighborhood for any node, for which however the products $\pi_{\min}N$ and $\pi_{\max}N$ do not lie in intervals bounded in $N$.

In order to apply Theorem 4 on growing families of graphs, we must make a further assumption ensuring that the products $\pi_{\min}N$ and $\pi_{\max}N$ are bounded from above and from below.

Even if this could seem a strong assumption, there are many cases of interest in which it is implicitly satisfied. A first case of course is when we have a family of symmetric consensus matrices, which can be produced using Metropolis-Hastings weights. Another simply implementable case is when one has a growing family of graphs $\mathcal{G}(N)$ and builds the $u$-th row of the corresponding consensus matrix $P(N)$ simply giving to each neighbor of $u$, including $u$ itself, the same, uniform weight

$$P_{uv}(N) = \begin{cases} \frac{1}{\delta_u}, & (u,\, v) \in E \\ \frac{1}{\delta_u}, & u = v \\ 0, & \text{otherwise} \end{cases}$$

where $\delta_u(N)$ is the number of in-neighbors of $u$ in the graph $\mathcal{G}(N)$. If $\delta_u(N) \leq \delta$ for any node $u$ and for any $N$, then it is possible to show that if $\pi(N)$ is the left invariant normalized eigenvector of $P(N)$, then all its entries lie in the interval $[\frac{1}{\delta^2 N},\ \frac{1}{4N}]$, thus the assumption is easily satisfied.

### 2.3.3 Applications to geometric graphs

In the previous paragraphs we have shown how to rewrite the performance cost of a consensus matrix $P$ we are interested in terms of average effective resistance of the associated graph $\mathcal{G}$. By using the properties of the effective resistances, it is also possible to show how to reduce some complex graph to more regular or simple graphs. In this section we are going to prove, as an example, that a class of geometric graphs behaves in the same manner as suitable regular Cayley graphs. The advantage is that for regular grids it is well known (see Proposition 2) how the cost depend on the number of nodes in the case of large graphs.

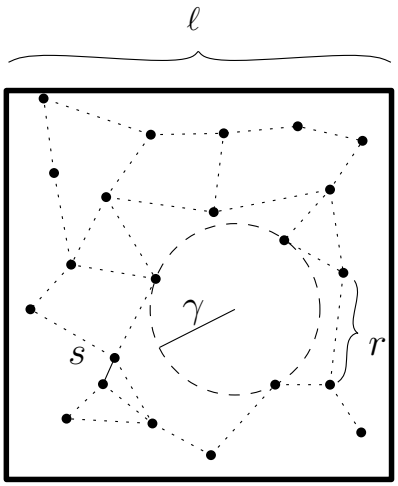


**Figure 11. An example of geometric graph, for which sharp values for $s$, $r$ and $\gamma$ are shown.**

### Geometric graphs: definitions and properties

Let $\mathcal{G} = (V,\, E)$ be a connected undirected graph such that $V \subset \mathbb{R}^d$ and $|V| = N$. Namely, the graph is deployed in some Euclidean space $\mathbb{R}^d$ and the nodes occupy some specific point in it. We assume that there exist an hypercube $Q \subset \mathbb{R}^d$ of edge length $\ell$ such that $V \subset Q$, namely all the nodes lie in the hypercube $Q$.

Following [5], we define following parameters:

- a lower bound on the minimum Euclidean distance between any two nodes:

$$s \leq \inf_{u,\, v \in V,\, u \neq v} \{d_{\mathrm{E}}(u,\, v)\}\,; \tag{24}$$

- an upper bound on the maximum Euclidean distance between any two connected nodes:

$$r \geq \sup_{(u,\, v) \in \mathcal{E}} \{d_{\mathrm{E}}(u,\, v)\}\,; \tag{25}$$

- an upper bound on the radius, $\gamma$, of the largest ball centered in $Q$ not containing any node of the graph:

$$\gamma \geq \max\{r |\, B(x,\, r) \cap V = \emptyset,\, \forall\, x \in Q\}\,; \tag{26}$$

- a lower bound on the minimum ratio between the Euclidean distance of two nodes and their graphical distance,

$$\rho \leq \min\left\{ \frac{d_{\mathrm{E}}(u,\, v)}{d_{\mathcal{G}}(u,\, v)} |\, (u,\, v) \in V \times V \right\}. \tag{27}$$

An example of such a graph in 2-D is drawn in Fig. 11, where the parameter $\rho$ is not specified.

Take now a growing family of geometric graphs $\mathcal{G}(N)$ deployed in $\mathbb{R}^d$, and assume that each of them respect the parameters $s$, $r$, $\gamma$ and $\rho$ as defined above. Take a family of reversible consensus

matrices $P(N)$, $P(N)$ associated with $\mathcal{G}(N)$, and call $\pi^T(N)$ the corresponding left invariant normalized eigenvector. Assume that all the nonzero entries of $P(N)$ lie in the interval $[p_{\min}, p_{\max}]$, and that any entry of $\pi^T(N)$, for any $N$ is such that $\beta_l \leq \pi_u^T(N) \leq \beta_u$. Then there exist two constants $C'_d$, $C''_d$, dependent on $p_{\max}$, $p_{\min}$, $\delta$, $d$, parameters of the geometric graph and $\beta_l$ and $\beta_u$, such that

- if $d = 1$,
$$C_1 N \leq J(P(N)) \leq C'_1 N\,;$$
- if $d = 2$,
$$C_2 \log N \leq J(P(N)) \leq C'_2 \log N\,;$$
- if $d \geq 3$,
$$C_d \leq J(P(N)) \leq C'_d\,.$$

# 3 Network management design

We studied the performance (in the sense of estimation error covariance) of real-time filtering running over two of the most promising routing protocols: Directed Staged Flooding (DSF) and Unicast Path Diversity (UPD), which is a specific implementation of a protocol based on TSMP [16]. In particular, we shown how to derive the end-to-end packet loss latency and connectivity statistics in terms of $\lambda_h^{ab}$, which is the probability that a packet sent from sensor $a$ is delivered to sensor $b$ with a delay $\tau$ not greater than $h$ (i.e. $\lambda_h^{ab} = \mathbf{P}[\tau \leq h]$). These statistics are used off-line to compute the performance of a Kalman-like estimator with a buffer of dimension $N$ storing the various measurements. These types of filters have been proposed in [42], and here we extend them to consider a shifted buffer, i.e. only measurements with a delay between $M$ and $M+N$, where $M$ is the buffer shift. Through numerical simulations, we shown that there is a trade off between performance, computational complexity and system dynamics, which might lead to regimes where one routing protocol is better than the other and regimes where the opposite occurs. This implies that protocols must be chosen with the specific application in mind.

## 3.1 Networking protocols for WSN: UPD and DSF

This section provides brief descriptions and Markov Chain models of two mesh routing protocols designed to provide high reliability for industrial control applications. For more details and examples, see [16].

### 3.1.1 Unicast Path Diversity

Dust Networks, Inc. proposed Unicast Path Diversity (UPD) over Time Synchronized Mesh Protocol (TSMP) [22], which exploits frequency, time, and space diversity for reliable networking in sensor networks. UPD is a many-to-one, multi-path routing protocol where each node in the network has multiple parents and the routing graph has no cycles. The links selected for routing are bidirectional, hence every link transmission can be acknowledged. If a packet transmission is not acknowledged, it is queued in the node for retransmission. To schedule the network, time is divided into time slots, and grouped into superframes. At each time slot, pairs of nodes are scheduled for transmitting a packet on different frequencies. The superframe containing the schedule of transmissions is repeated over time. Our model uses frequency hopping to justify the assumption that links are independent over retransmissions.

In order to calculate $\lambda_h^{ab}$, we construct a general Mesh TDMA Markov Chain (MTMC) model for UPD that assumes knowledge of the routing topology, schedule, and all the link probabilities. MTMC models single packet transmission in the network without the effects of queuing.

*Mesh TDMA Markov Chain Model:* let us represent the routing topology as a graph $G = (\mathcal{V}, \mathcal{E})$, and denote a node in the network as $i \in \mathcal{V} = 1, \ldots, N$, and a link in the network as $l \in \mathcal{E} \subset \{(i,j) | i,j \in \mathcal{V}\}$, where $l = (i,j)$ is a link for transmitting packets from node $i$ to node $j$. Time $h$ will be measured in units of time slots, and let $H$ denote the number of time slots in a superframe. The link success probability for link $l = (i,j)$ at time slot $h$ is denoted $p_l^{(h)}$, or $p_{ij}^{(h)}$. We set $p_l^{(h)} = 0$ when link $l$ is not scheduled to transmit at time $h$. It is possible to construct the following time-varying, discrete-time Markov Chain:

**Definition 4** (MTMC Model)**.** *Let the set of states in the Markov Chain be the nodes in the network, $\mathcal{V}$. The transition probability from state $i$ to state $j$ at time $h$ is simply $p_{ij}^{(h)}$, with $p_{ii}^{(h)} = 1 - \sum_{j \neq i} p_{ij}^{(h)}$.*

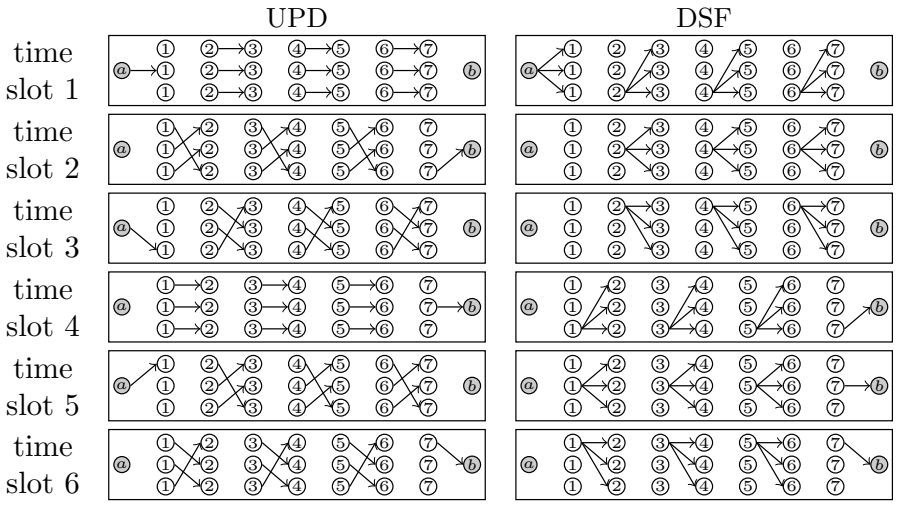


**Figure 12.** (left) UPD and (right) DSF schedules for routing on a grid of width 3, used in the calculations for the graph in Fig. 13.

*Let* $\mathbf{P}^{(h)} = [p_{ij}^{(h)}]^T \in [0,1]^{N\times N}$ *be the column stochastic transition probability matrix for a time slot and* $\mathbf{P}^{(\underline{H})} := \mathbf{P}^{(H)}\mathbf{P}^{(H-1)}\ldots\mathbf{P}^{(1)}$ *be the transition probability matrix for a repeating superframe. Let a packet originated at node* $a$ *be represented by* $\mathbf{p}^{(0)} := \mathbf{e}^{[a]}$*, where* $\mathbf{e}^{[a]}$ *is an elementary vector with the* $a$*-th element equal to 1 and all other elements equal to 0. The probability distribution of the state at time* $h$ *given the initial condition* $\mathbf{e}^{[a]}$ *is given by* $\mathbf{p}^{(h)} = \left(\prod_{t=1}^{h}\mathbf{P}^{(t)}\right)\mathbf{p}^{(0)}$*. With this definition the probability that a packet is delivered from* $a$ *to* $b$ *with a delay not greater than* $h$ *is* $\lambda_h^{ab} = \mathbf{p}_b^{(h)}$ *(the* $b$*-th element of* $\mathbf{p}^{(h)}$*).*

### 3.1.2 Directed Staged Flooding

Directed Staged Flooding (DSF) uses simple constrained flooding for one-to-many or one-to-one routing. *Unlike* UPD, DSF provides increased end-to-end connectivity with less latency by multicasting packets instead of using acknowledgments and retransmissions. *Like* UPD, DSF assumes that the nodes follow a TDMA routing schedule. During a transmission each node transmits to a subset of its neighboring nodes. Furthermore, we group the nodes along the end-to-end transmission path such that a packet is modeled as being passed between groups of nodes, and we call each group of nodes a *stage*. For instance, looking at the node topology for one time slot on the right of Fig. 12, each column of nodes is a stage. For simplicity, here we assume the nodes are not shared between stages, although this is not required (see [16]).

We use a Directed Staged Flooding Markov Chain (DSFMC) model to calculate $\lambda_h^{ab}$, assuming the routing schedule, the stage grouping, and all the link probabilities are known. The model requires the *sets* of link transmissions between *distinct* pairs of stages to be independent. Like UPD, DSF uses frequency hopping over time to help justify this assumption. However, the model allows the link transmissions between the *same* pair of stages to be correlated.

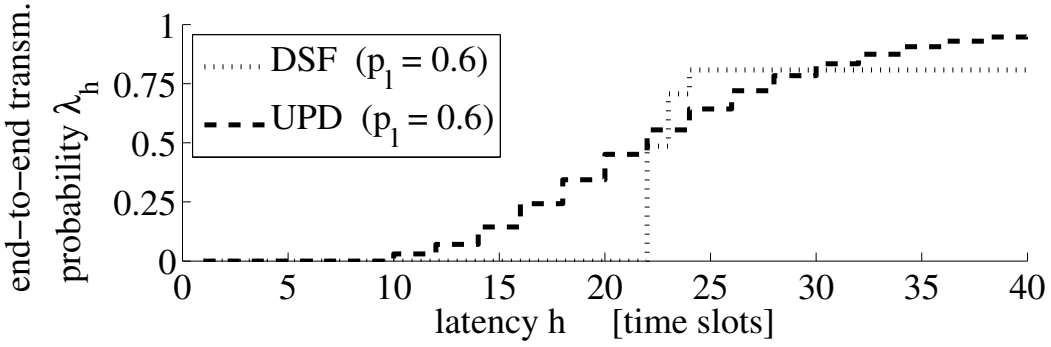


**Figure 13.** *Calculated* **end-to-end connectivity $\lambda_h^{ab}$ as a function of latency $h$ using the routing schedules described in Fig. 12, where all the links have probability $p_l = 0.6$, $a = 0$ and $b = 8$.**

*Directed Staged Flooding Markov Chain Model:* The notation for the routing topology is the same as in the MTMC model. Note that here the link success probability for link $l = (i, j)$ is treated as being time-invariant and is denoted $p_l$ (or $p_{ij}$), since each link is used only once when transmitting a single packet.

It is possible to construct a time-invariant, discrete-time Markov Chain where the state at a stage represents the *set* of nodes that successfully received a copy of the packet. The transition probabilities between the states depend on the joint probability of successful link transmissions between stages. When the links are all independent, the model is:

**Definition 5** (DSFMC Model)**.** *Let's assume we have a routing topology with $K+1$ stages $0, \ldots, K$. Each stage $k$ has $N_k$ nodes, and the set of $2^{N_k}$ possible states in stage $k$ is represented by the set of numbers $\mathcal{S}^{(k)} = \{0, \ldots, 2^{N_k} - 1\}$. Let $\mathcal{K}^{(k)}$ be the set of nodes in stage $k$ and for each state $\sigma^{(k)} \in \mathcal{S}^{(k)}$, let $\mathcal{R}_\sigma^{(k)} \subset \mathcal{K}^{(k)}$ be the set of nodes that have received a copy of the packet and $\mathcal{U}_\sigma^{(k)} = \mathcal{K}^{(k)} \backslash \mathcal{R}_\sigma^{(k)}$ be the set of nodes that have not received a copy of the packet (see Fig. 14). Let $\omega^{(k)}$ denote the state where no nodes received a copy of the packet in stage $k$. The conditional probability that the next state $\mathbf{X}^{(k+1)}$ equals $\sigma^{(k+1)}$ given that the current state $\mathbf{X}^{(k)}$ equals $\omega^{(k)}$ is $0$ if $\sigma^{(k+1)} \neq \omega^{(k)}$, and $1$ if $\sigma^{(k+1)} = \omega^{(k)}$. If $\sigma^{(k)} \neq \omega^{(k)}$:*

$$\mathcal{P}[\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)}] = \left( \prod_{\substack{u \in \mathcal{U}_\sigma^{(k+1)} \\ i \in \mathcal{R}_\sigma^{(k)}}} (1 - p_{iu}) \right) \prod_{r \in \mathcal{R}_\sigma^{(k+1)}} \left( 1 - \prod_{i \in \mathcal{R}_\sigma^{(k)}} (1 - p_{ir}) \right)$$

*The transition probability matrices between stage $k$ and $k+1$ are $\mathbf{P}^{(k+1)} \in [0,1]^{N_{k+1} \times N_k}$, where the entry in position $(\sigma^{(k+1)}, \sigma^{(k)})$ of the matrix is $\mathcal{P}[\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)}]$. The initial state $\mathbf{X}^{(0)}$ is the state $\sigma^{(0)}$ corresponding to $\mathcal{R}_\sigma^{(0)} = \{a\}$, where $a$ is the node sending the initial packet. Then, the probability distribution $\mathbf{p}^{(k)} \in [0,1]^{N_k}$ of the state at stage $k$ is $\mathbf{p}^{(k)} = \left( \prod_{t=1}^{k} \mathbf{P}^{(t)} \right) \mathbf{p}^{(0)}$. Assume the transmissions of nodes within a stage must be scheduled in separate time slots so they do not interfere with each other, so time $h$ is related to stage $k$ by $h = \sum_{j=0}^{k-1} N_j$. Then, $\lambda_h^{ab} = \sum_{\{\sigma^{(k)} | b \in \mathcal{R}_\sigma^{(k)}\}} \mathcal{P}[\mathbf{X}^{(k)} = \sigma^{(k)}]$, a summation over the corresponding elements in the vector $\mathbf{p}^{(k)}$.*

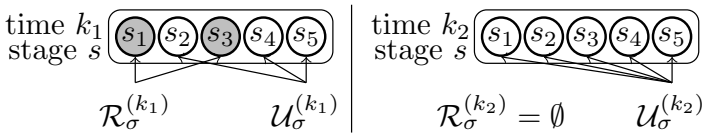


**Figure 14. Mapping of states to nodes that received a packet in the DSFMC model. A state $\sigma^{(k)}$ of a stage is in correspondence with the set of nodes $\mathcal{R}_\sigma^{(k)}$ that correctly received the packet (greyed circles).**

### 3.1.3 UPD and DSF Comparison

UPD can deliver packets from $a$ to $b$ in a shorter time than DSF, but with a larger variance. Also $\lim_{h\to\infty} \lambda_h^{ab} = 1$ for UPD, while $\lambda_h^{ab} \leq 1$ for DSF after the last stage transmits (assuming $p_l \neq 1$). This imply that UPD can always provide better end-to-end connectivity at high latencies $h$. DSF tends to perform better when there are a few very poor links scattered throughout the network. Fig. 13 compares $\lambda_h^{ab}$ for various $h$ for UPD and DSF under the schedules in Fig. 12, assuming $p_l = 0.6 \ \ \forall l \in \mathcal{V}$.

### 3.1.4 Usage of UPD and DSF for estimation

Both protocols can be used for estimation purposes. Assume that nodes $a = 0$ and $b = 8$ in Fig.13 are respectively a sensor and an estimator; $a$ collects data and then sends packets towards $b$ through the UPD or DSF network. How should $b$ handle packet loss and delay to have some kind of optimal estimate? Which protocol behaves better and under what conditions? In the next sections we will formally state the problem assuming that $a$ measures the following discrete time linear stochastic plant:

$$x_{t+1} = Ax_t + w_t \qquad\qquad y_t = Cx_t + v_t \tag{28}$$

where $t \in \mathbb{N}$, $x, w \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, $y, v \in \mathbb{R}^m$, $C \in \mathbb{R}^{m\times n}$, $(x_0, w_t, v_t)$ are Gaussian, uncorrelated, white, with mean $(\bar{x}_0, 0, 0)$ and covariance $(P_0, Q, R)$ respectively (note $P \neq \mathbf{P}$ from Defs. 4 and 5). We also assume that the pair $(A, C)$ is observable, $(A, Q^{1/2})$ is reachable, and $R > 0$.

Note that measurements are time-stamped, encapsulated into packets, and then transmitted through the digital communication network. Time-stamping of measurements is necessary to reorder packets at the receiver side since they can arrive out of order.

## 3.2 Minimum variance estimators subject to packet loss and delay

The packet arrival process can be modeled via the random variable $\gamma_k^t$, defined as follows ($k$ is transmit time):

$$\gamma_k^t = \begin{cases} 1 & \text{if } y_k \text{ is received at or before time } t,\ t \geq k \\ 0 & \text{otherwise} \end{cases}$$

We also define the packet delay $\tau_k \in \{\mathbb{N}, \infty\}$ for observation $y_k$ as follows:

$$\tau_k = \begin{cases} \infty & \text{if } \gamma_k^t = 0, \forall t \geq k \\ t_k - k & \text{otherwise } (t_k := \min\{t | \gamma_k^t = 1\}) \end{cases} \tag{29}$$

where $t_k$ is the arrival time of observation $y_k$ at the estimator. If the delay of the arriving packets is bounded, i.e. if there exists $\bar{N}$ such that $\gamma_k^t = \gamma_k^{t+1}$ for $t-k+1 \geq \bar{N}$, then it has been shown in [42] that the minimum variance estimator $\hat{x}_{t|t}^t = \mathbb{E}[x_t \,|\, \text{arrived measurements}] = \mathbb{E}[x_t \,|\, \gamma_1^t, .., \gamma_t^t, \tilde{y}_1^t, .., \tilde{y}_t^t]$ (where $\tilde{y}_k^t = \gamma_k^t y_k$) and its corresponding prediction error covariance $P_{t+1|t}^t = \mathbb{E}[(x_{t+1} - A\hat{x}_{t|t}^t)(x_{t+1} - A\hat{x}_{t|t}^t)^T \,|\, \gamma_1^t, \ldots, \gamma_t^t]$ is given by a time-varying Kalman Filter with a buffer of size $\bar{N}$ whose equations are:

$$\begin{aligned}
\hat{x}_{t-\bar{N}|t-\bar{N}}^t &= \hat{x}_{t-\bar{N}|t-\bar{N}}^{t-1}, \quad P_{t-\bar{N}+1|t-\bar{N}}^t = P_{t-\bar{N}+1|t-\bar{N}}^{t-1} \\
\hat{x}_{k|k}^t &= A\hat{x}_{k-1|k-1}^t + \gamma_k^t K_k^t (\tilde{y}_k^t - CA\hat{x}_{k-1|k-1}^t) \\
K_k^t &= P_{k|k-1}^t C^T (CP_{k|k-1}^t C^T + R)^{-1} \\
P_{k+1|k}^t &= AP_{k|k-1}^t A^T + Q - \gamma_k^t AK_k^t CP_{k|k-1}^t A^T
\end{aligned}$$

where $k = t - \bar{N} + 1, \ldots, t$, and $\hat{x}_{h|h}^t = \bar{x}_0, P_{h|h-1}^t = P_0$ for $h \leq 0$. Because the error covariance $P_{t+1|t}^t$ depends on the packet arrival sequence $\gamma_k^t$, it is time-varying and does not converge to a steady state, unlike the standard Kalman Filter with no packet loss. Moreover, it requires the inversion of up to $\bar{N}$ matrices at every time step $t$ and might be too expensive for on-line implementation. Also, the buffer size $\bar{N}$ needed for the optimal estimator might be too large. Although in theory even very old measurements help reduce the estimation error, in practice their contribution is marginal. In Sec. 3.3 we propose a new strategy requiring no matrix inversion and whose buffer size can be reduced to trade off performance with computational complexity.

### 3.3 Estimation with shifted buffer and constant gains

In this section we propose a suboptimal estimator design strategy which does not require any matrix inversion and has a buffer with length smaller than the WSN maximum packet delay $\bar{N}$. Since we want to quantify the performance of the estimator, we need to specify the statistics of the packet arrival process from sensor $a$ to estimator $b$. We assume it to be stationary and i.i.d. with probability function (in the following we will omit the superscripts): $\lambda_h := \lambda_h^{ab} = ¶[\tau_t \leq h]$, where $t \geq 0$, $0 \leq \lambda_h \leq 1$ is non-decreasing in $h = 0, 1, \ldots, \bar{N}$, and $\tau_t$ was defined in Eqn. (29). Although arrivals might not be i.i.d. because of correlation in packet delays, the i.i.d. assumption allows us to explicitly compute the performance of the proposed estimators and to find the optimal gains within this class.

Starting from the buffer of the optimal filter described in Sec. 3.2, we consider the subset of the measurements with time delays in $M, \ldots, M+N$ (the subset will be called a *shifted buffer*), where $M = 0, \ldots, \bar{N}$ is the starting point of the shifted buffer, and $N = 1, \ldots, \bar{N} - M$ is its length (an example is shown in Fig. 15). The estimation scheme has the following structure:

$$\begin{aligned}
\tilde{x}_{t-M-N|t-M-N}^t &= \hat{x}_{t-M-N|t-M-N}^{t-1} \\
\tilde{x}_{k|k}^t &= A\tilde{x}_{k-1|k-1}^t + \gamma_k^t \tilde{K}_{t-k} (\tilde{y}_k^t - CA\tilde{x}_{k-1|k-1}^t), \\
\tilde{x}_{t|t}^t &= A^M \tilde{x}_{t-M|t-M}^t
\end{aligned} \tag{30}$$

where $k = t-M-N+1, \ldots, t-M$, which mimics the time-varying estimator with the buffer in the previous section, but with gains $\{\tilde{K}_h\}_{h=M}^{M+N-1}$ not depending on the packet arrival sequence $\gamma_k^t$, unlike the gains $\{K_k^t\}$ of the optimal filter of Sec. 3.2.

The performance of this new estimator is measured in terms of its prediction error covariance $\tilde{P}_{t+1|t} = \mathbb{E}[(x_{t+1} - A\tilde{x}^t_{t|t})(x_{t+1} - A\tilde{x}^t_{t|t})^T \,|\, \gamma^t_1, ..., \gamma^t_t]$. Obviously it must be $P^t_{t+1|t} \leq \tilde{P}^t_{t+1|t}$ for every sequence $\gamma^t_k$ since the filter in the previous section is the minimum variance linear filter. Just like $P^t_{t+1|t}$, the prediction error covariance $\tilde{P}^t_{t+1|t}$ is a random variable since it depends on the specific realization of the arrival process $\gamma^t_k$. Therefore, we are interested in computing the expected prediction error covariance with respect to all possible realizations of $\gamma^t_k$, i.e. $\overline{P}^t_{t+1|t} = \mathbb{E}_\gamma[\tilde{P}^t_{t+1|t}] = \overline{P}^t_{t+1|t}(\tilde{K}, N, M)$, where we made explicit the dependence on the gains $\tilde{K} = (\tilde{K}_M, \ldots, \tilde{K}_{M+N-1})$, the length of the buffer $N$, and its initial position $M$. The following theorem provides stability conditions for the proposed filter.

**Theorem 6.** *Consider the following modified A.R.E.:*

$$P = \Phi_\lambda(P) = APA^T + Q - \lambda APC^T(CPC^T + R)^{-1}CPA^T \tag{31}$$

*and the gain $K_P = g(P) = PC^T(CPC^T + R)^{-1}$. If $A$ is unstable, then there exists a unique positive semidefinite solution if and only if $\lambda > \lambda_c$ where:*

- *$\lambda_c$ depends only on the pair $(A, C)$;*
- *$\lambda_c$ satisfies the following inequalities (where the $\sigma^u_i(A)$'s are the unstable eigenvalues of $A$):*

$$p_{min} = \frac{1}{\prod_i |\sigma^u_i(A)|^2} \leq 1 - \lambda_c \leq \frac{1}{\max_i |\sigma^u_i(A)|^2} = p_{max}$$

- *$p_{min} = 1 - \lambda_c$ if $C$ is rank one;*
- *$p_{max} = 1 - \lambda_c$ if $C$ is invertible.*

*If $A$ is strictly stable, then there always exists a unique positive semidefinite solution. Consider also the class of filters defined by Eqn.* (30)*, and suppose the packet arrival process is i.i.d.. If $\lambda_{M+N-1} < \lambda_c$ then $\lim_{t\to\infty} \sup_t \overline{P}^t_{t+1|t}(\tilde{K}, N, M) = \infty$ for any choice of the gains $\tilde{K}$. If $\lambda_{M+N-1} > \lambda_c$, then consider the following positive semidefinite matrices:*

$$\begin{aligned} &V_{M+N-1} = \Phi_{\lambda_{M+N-1}}(V_{M+N-1}) \\ &V_k = \Phi_{\lambda_{k+1}}(V_{k+1}), \quad k = M+N-2, \ldots, M \\ &V_k = AV_{k+1}A^T + Q = \Phi_0(V_{k+1}), \quad k = M-1, \ldots, 0 \end{aligned} \tag{32}$$

*and the gains $\tilde{K}^*_k = g(V_k), k = M+N-1, \ldots, M$. Then:*

$$\begin{aligned} &\lim_{t\to\infty} \overline{P}^t_{t+1|t}(\tilde{K}^*, N, M) = V_0(N, M) \\ &\lim_{t\to\infty} \overline{P}^t_{t+1|t}(\tilde{K}, N, M) \geq V_0(N, M), \quad \forall \tilde{K} \end{aligned} \quad .$$

*Finally $V_0(N, M) \geq V_0(N+1, M)$.*

Thm. 6 states that if the packet arrival probability for the last slot in the buffer $\lambda_{M+N-1}$ is sufficiently high, then there exists a stable estimator within the class of filters here proposed. Thm. 6 also shows how to find the best estimator in terms of minimum variance within this class. The best expected prediction error covariance $V_0(N, M)$ is a function of the buffer length $N$ and initial position $M$. The memory and computational complexity for such estimators do not depend on $M$. Therefore, we would like to

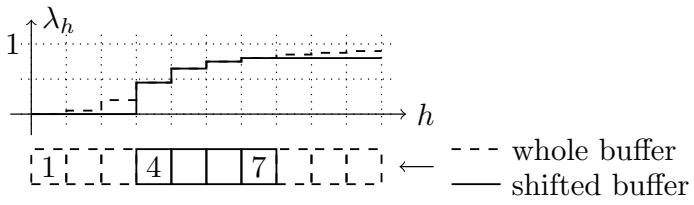


**Figure 15. Example of shifted buffer with $M=3$ and $N=4$; the elements of the buffer and the $\lambda_h$'s used in Eqn. (31) are plotted with a continuous line. The dashed $\lambda_h$ refers to the trivial buffer with $M=0$ and $N=\bar{N}$.**

find the best $M$ which minimizes $V_0(N,M)$. Unfortunately it is not possible to guarantee that there exits $M^*$ such that $V_0(N,M^*) \leq V_0(N,M)$, and indeed this is actually false in general. To overcome this limitation, we will consider a cost function which is linear and positive in $V \geq 0$, i.e. a function $f : \mathbb{R}^{n\times n} \to \mathbb{R}^+$. Some examples are $f(V) = \mathrm{trace}(V)$ and $f(V) = z^T V z$, where $z \in \mathbb{R}^n$. Using this cost function we will compute the optimal shifted buffer $M$ for any fixed $N$ as:

$$M^*(N) = \arg\min_M f\left(V_0(N,M)\right)$$

and the corresponding minimum cost $v^*(N) = \min_M f(V_0(N,M))$. Since $M$ is an integer, it is not possible to find the minimum in closed form. Therefore, we need to explicitly compute $f(V_0(N,M))$ for all $M$. However, this can be done off-line and then used for on-line estimation.

### 3.4 Estimation performance under UPD and DSF routing protocols

In this section we apply the results of Sec. 3.2 to the situation proposed in Sec. 3.1.4 to evaluate performances for the 2D target tracking application. A popular model for the dynamics of a moving target is given by a double integrator subject to white noise, i.e. $\ddot{\xi}_x(t) = w_x(t)$ where $\xi_x$ is the position of the moving target along the $x$-axis and $w_x(t)$ is continuous time white noise with zero-mean and variance $q$. We also assume that the position measure is noisy, i.e. $y_x(t) = \xi_x(t) + v(t)$, where $v(t)$ is zero mean white noise with variance $I$. The dynamics along the $y$-axis are modeled similarly and the noises are assumed to be uncorrelated along the two axes. The state space dynamics, discretized with period T, are written as:

$$x_{t+1} = \begin{bmatrix} G & 0 \\ 0 & G \end{bmatrix} x_t + w_t, \quad y_t = \begin{bmatrix} H & 0 \\ 0 & H \end{bmatrix} x_t + v_t$$
$$G = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \; H = \begin{bmatrix} 1 & 0 \end{bmatrix}, \; S = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix}$$

where $x^T = [\xi_x \, \dot{\xi}_x \, \xi_y \, \dot{\xi}_y]$, $w_t$ and $v_t$ are white Gaussian noise with covariance $Q = q \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix}$ and $R = rI$ respectively, and $I$ is the identity matrix. In this case $(A, Q^{1/2})$ is reachable, $(A, C)$ is observable and the critical packet arrival probability introduced in Thm. 6 is $\lambda_c = 0$ (all the eigenvalues are one).

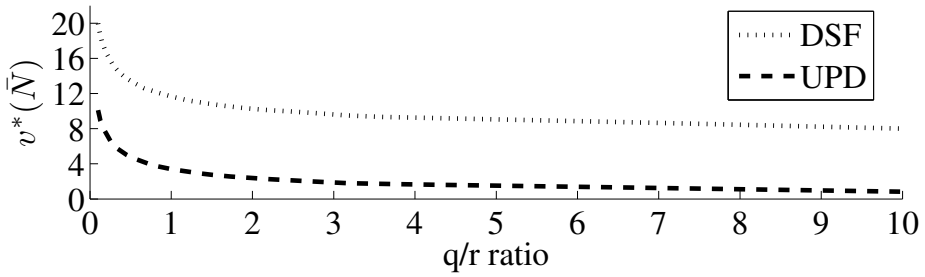


**Figure 16. Estimation error cost $v^*(\bar{N})$ for the full length buffer as a function of the ratio $q/r$ with $q = 1$ for the UPD and DSF protocols.**

Since the behavior of the filter is regulated by the ratio $q/r$, we fix w.l.o.g. $q = 1$ and evaluate the performance as a function of $r$, in terms of the mean square prediction error

$$v^*(N) = f(V_0(N, M^*(N)) = z^T V_0(N, M^*(N))z$$

on the position of the moving target, where $z^T = [1\,0\,1\,0]$ and $V_0(N, M)$ is the expected prediction error covariance of the estimator with a shifted buffer with size $N$ and initial position $M$ defined in Sec. 3.3.

First we compute the best achievable performance of filters with constant gains as a function of $r$ for UPD and DSF protocols with end-to-end packet delay statistics shown in Fig. 13. Noting that $\lambda_h^{\text{UPD}} = 0$ for $h < 10$ and $\lambda_h^{\text{UPD}} \approx 1$ for $h > 40$ we can set $\bar{N}^{\text{UPD}} = 40$, i.e. almost all packets arrive with a delay between 10 and 40 time steps. On the other hand $\lambda_h^{\text{DSF}} = 0$ for $h < 22$ and $\lambda_h^{\text{DSF}} = \lambda_{h+1}^{\text{DSF}} = 0.81$ for $h \geq 24$, which implies that $\bar{N}^{\text{DSF}} = 24$ and that packet loss probability is $p_{loss}^{\text{DSF}} = 1 - 0.81 = 0.19$.

Fig. 16 compares the best achievable performance in terms of $v^*(\bar{N})$ and shows that UPD always performs better than DSF for the topology and link probabilities of Fig. 12. This is to be expected for the two extreme regimes, i.e. for large $q/r$ and for small $q/r$. In fact, for large $q/r$, old measurements cannot reduce the estimation error since $x_t$ changes too rapidly. Since UPD delivers some packets with much smaller delay than DSF, it should perform better. For small $q/r$, $x_t$ changes very slowly, so old measurements reduce the estimation error. Therefore, a relevant parameter is the packet loss probability, which is bigger for DSF. Note that UPD performs better for all the $q/r$ ratios for *this* particular topology and link probabilities. It can be shown that in other cases the situation can be inverted.

When using a buffer of size $N < \bar{N}$ there are tradeoffs between estimation and computational complexity. Fig. 17 shows the performance of the filters as a function of $N$ for three different noise ratios $q/r$ (buffer shift $M$ being chosen optimally $\forall N$). As expected, the performance for DSF becomes constant for $N > 3$ since all packets arrives with delay $h \in \{22, 23, 24\}$. Instead the performance of UPD improves until $N = 30$ (range of delay of the packets), and after becomes constant; anyway the improvements after $N > 20$ are so small that it is not useful to use longer buffers. Note that, if both $q/r$ and the buffer are very small, DSF performs better than UPD. Therefore, it is not possible to claim that UPD is always superior to DSF.

Finally, Fig. 18 shows the optimal buffer shift $M^*(N)$ as a function of the buffer length. As expected, $M^*_{\text{DSF}}(N)$ is around the minimum delay of the measurements (i.e. 22). For $N > 3$, the performance becomes constant, and the optimal $M^*_{\text{DSF}}(N)$ is not unique, since any buffer including delays $h \in$

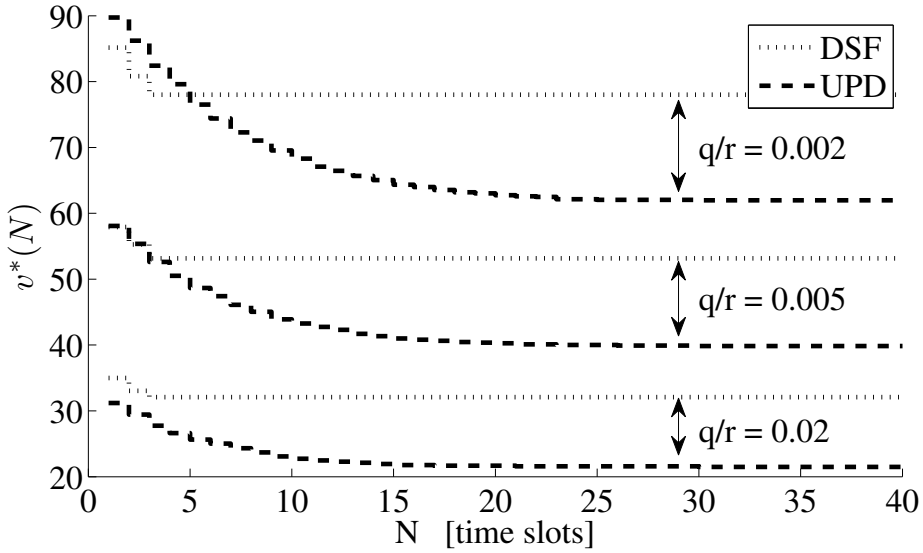


**Figure 17. Estimation error cost $v^*(N)$ relative to the optimal buffer shift $M^*(N)$ as a function of the buffer length $N$. The functions are plotted for three different ratios of $q/r$ and for both UPD and DSF.**

$\{22, 23, 24\}$ performs optimally. Therefore $M^*_{\text{DSF}}(N)$ s.t. $N > 3$ can be chosen such that $M^{D*}_{min}(N) \leq M^*_{\text{DSF}}(N) \leq M^{D*}_{max}(N)$. The UPD case depends more on the $q/r$ ratio. When $N$ is small we will use a few recent measurements, while when $N$ is large we will include more older measurements. In the case of Fig. 18, $M^*_{\text{UPD}}$ initially decreases as $N$ increases, indicating that the buffer adds packets with smaller delays. Then for $h < 10$ the buffer adds the packets with bigger delays (a further decrease in $M$ does not help). When $N > 30$, the addition of packets with smaller delay also provides negligible improvements in performance so each $M^*_{\text{UPD}}(N)$ can be chosen s.t. $M^{U*}_{min}(N) \leq M^*_{\text{UPD}}(N) \leq M^{U*}_{max}(N)$.

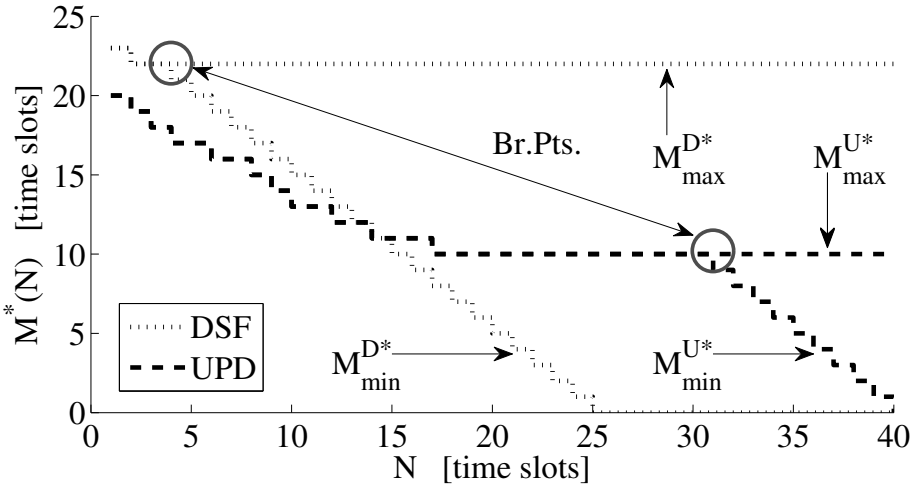


**Figure 18. Optimal buffer shift $M^*(N)$ as a function of the buffer length $N$ for $q = 1$ and $r = 0.005$. After the branching point (Br.Pts. in figure), the value $M^*(N)$ can be any point between the two branches, i.e. $M^*_{min}(N) \leq M^*(N) \leq M^*_{max}(N)$.**

# 4 Network coding design

In this section we will propose in more details the coding techniques for consensus algorithms in the case of noiseless digital channels, presented in [4], and in case of noisy digital channels, presented in [10].

## 4.1 A coding method for consensus in case of noiseless digital channels

In recent years, motivated by the possible great diffusion of wireless networks, researchers have addressed their efforts in finding algorithms able to solve specific problems in a distributed way. Decentralized control of autonomous vehicles and distributed Kalman filtering are two examples of what is potentially achievable exploiting, with a suitable algorithm, all the capabilities of a wireless network.

A common feature of these algorithms is that they have to take into account many constraints on the information flow, since the agents can exchange information through some communication network. Such a network will be hereafter modelled as a directed graph $\mathcal{G} = (V, \mathcal{E})$, where $V$ is the set of agents $V = \{1, \ldots, N\}$ while $\mathcal{E}$, the set of edges, is a subset of $V \times V$ such that $(i, j) \in \mathcal{E}$ iff the agent $j$ can send information to the agent $i$.

One of the simplest problem for which a distributed algorithm have been found is the so called average–consensus problem, where a network of interconnected agents is required to compute the mean of some numbers. The first approach appeared in the literature (see, i.e. [37]) modelled every agent's state as a real value $x_i(t)$, and set the evolution in time accordingly to these difference equations

$$x_i(t+1) = \sum_{i=1}^{N} p_{ij} x_j(t), \tag{33}$$

where $p_{ij}$ are coefficients complying with the communication constraints between agents, thus $p_{ij} \neq 0$ only if the edge $(i, j)$ belongs to $\mathcal{E}$. Equation 33 imply that every agent, during algorithm evolution, keeps update his state with a proper average among his state and those of his neighbours. More compactly we can write

$$x(t+1) = Px(t) = (I + K)x(t), \tag{34}$$

where $P = (p_{ij}) \in \mathbb{R}^{N \times N}$, $K = (k_{ij}) = P - I$ and $x(t) \in \mathbb{R}^N$ groups all agents states in a single state vector.

In [37] it is shown that, if the graph $\mathcal{G}$ is strongly connected, every irreducible doubly stochastic[3] matrix $P$ drives the system dynamic to the consensus, namely

$$\lim_{t \to \infty} x(t) = \frac{\mathbf{1^T x(0)}}{N} \mathbf{1}, \tag{35}$$

where $\mathbf{1} \in \mathbb{R}^{\mathbf{N}}$ is the vector of all ones.

This simply algorithm implicitly assumes also that the communication network is ideal, so that real numbers are exchanged between agents without any loss or degradation of information. Of course this assumption, in real applications, is not realistic due to energy and bandwidth limitations and, for this reason, a lot of literature has been devoted to investigate the effects of noise or packet drop on the algorithm performances (see i.e. [7] and [23]). Another limitation of the algorithm 34 is that it requires the agents to communicate each others their actual states. As already pointed out, these measures belong to $\mathbb{R}$ or, more generally, to a non countable set and, thus, cannot be sent through a digital channel in a finite time. A naive solution to this problem is to send information with a loss of precision by coding data with a fixed number of bit. Usually this solution is implemented using the same coding used by agents CPU's to encode real numbers which is generally the standard floating-point double precision (64 bits) encoding provided by IEEE.

This solution has two great disadvantages. The first one is related to exploitation of the communication channel which is generally unoptimized. One need only reflect on the fact that, when the consensus is almost reached, the agents keep sending whole words of 64 bits length while the information is, at that point, contained only in a few less significant bits. The second disadvantage is that the precision loss during communication, together with round–off errors affecting agents computations, cause the algorithm to converge to a value that could be slightly different from the initial mean of the states.

In [11] and [12] some analysis on quantization effects in standard consensus algorithm are presented as well as an improved algorithm able to achieve exact average–consensus using a finite number of bits every algorithm step. Unfortunately the estimation on the required number of bits provided in [12] increases as the number of agents grows despite simulations show a different trend. Motivated by such numerical results, in this paper we present a novel algorithm, to which we will refer as the Zoom-in Zoom-out algorithm, that can achieve exact convergence as well and whose parameters can be chosen, with some reasonable assumptions, regardless to the number of agents. Even if the theoretical finiteness of the number of bits required by this algorithm it is not yet proven, numerical results suggest that this number is finite, it depends only on the algorithm parameters and that it is independent of the number of agents.

[3] Under the assumption made on the graph, it is always possible to find such a matrix complying with communication constraints.

The idea behind ZIZO–algorithm is to slightly modify equation 34 in order to obtain a state evolution of the form

$$x(t+1) = x(t) + K\hat{x}(t), \tag{36}$$

where $\hat{x}(t)$ is a suitable estimation of the real state at time $t$.

From 36 it turns out that each agent should keep trace of his neighbours' state estimations and has to perform additional operations in order to keep them updated.

In detail, generic ith agent has to manage the following variables:

- $x_i(t)$, the state of the agent.
- $\hat{x}_{ij}(t)\,\forall j : (i,j) \in \mathcal{E}$, the estimations, made by the agent $i$, of his neighbour j state as well as the estimation of his own measure even if this could seems paradoxical.
- $z_{ij}(t)$, the zoom factors associated to the estimation $\hat{x}_{ij}(t)$, whose aim will be clearer later.

The algorithm has three positive parameters $k_1$, $k_2$ and $q$ as well as the matrix of coefficients $K$ which is supposed to be given and designed to obtain, in the linear case, a suitable convergence rate over a given graph. At every time $t$ the generic ith agent performs these steps:

1. It computes the quantization level $l_i$ related to his state $x_i$, namely
$$l_i(t) = \left\lfloor \frac{x_i(t) - \hat{x}_{ii}(t)}{q|z_{ii}(t)|} \right\rfloor + \frac{1}{2} \tag{37}$$
if $|z_{ii}(t)| \neq 0$. Of course $l_i(t)$ takes values only in a countable set.
2. It sends the level $l_i(t)$ to all his neighbours. This is the step which involves communication between agents.
3. It computes the quantities
$$f_{ij}(t) = l_j(t) q |z_{ij}(t)| \tag{38}$$
using the levels $l_j$ received from his neighbours.
4. It sets the variables for the next time step
$$\begin{cases} \hat{x}_{ij}(t+1) &= \hat{x}_{ij}(t) + f_{ij}(t) \\ x_i(t+1) &= x_i(t) + \sum_{j:(i,j)\in\mathcal{E}} k_{ij}\hat{x}_{ij}(t+1) \\ z_{ij}(t+1) &= k_1|z_{ij}(t)|\text{sgn}(f_{ij}(t)) + k_2 f_{ij}(t) \end{cases} \tag{39}$$
where $\text{sgn}(x) = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ -1 & \text{if} \quad x < 0 \end{cases}$.

It is clear from 37 and 38 that the magnitude of zoom factors afflict the state estimations accuracy. The bigger the zoom factors are, the roughly the estimations will be; this zoom–reliant estimation behaviour justify the name given to the algorithm. Since zoom factors play such a fundamental role, particular care was given to the design of their dynamics. More precisely in the third of 39, the term $k_2 f_{ij}(t)$ should guarantee the zoom factors to follow the difference between real states and estimations, thus improving the estimation quality as well as the estimations come close to the real states. The term $k_1|z_{ij}(t)|\text{sgn}(f_{ij}(t))$, on the other hand, prevent zoom factors from becoming too small in a single algorithm step as a consequence of a fortuitous coincidence between states and estimations

**Remark 2.** *Note that, from the algorithm equations, the following holds*

$$|z_{ij}(t)| = k_1|z_{ij}(t-1)| + k_2|f_{ij}(t)| \geq k_1|z_{ij}(t-1)|,$$

*thus, if* $z_{ij}(0) \neq 0$*, we have* $z_{ij}(t) \neq 0\ \forall t$ *and equation 37 is always well–posed.*

In the following part of this section we will look for a suitable nonlinear state–space system able to describe how agents' states and estimation variables evolve. For this purpose a useful simplification is given in the following proposition.

**Proposition 5.** *If the ZIZO–Algorithm is initially synchronized, that is* $\hat{x}_{ii}(0) = \hat{x}_{ji}(0)\ \ z_{ii}(k) = z_{ji}(0)\ \forall (i,j) \in \mathcal{E}$*, then it stays synchronized.*

Under the hypothesis of proposition 5, the whole network state is described by $3n$ variables, namely $x_i$, $\hat{x}_i = \hat{x}_{ii} = \hat{x}_{ji}$ and $z_i = z_{ii} = z_{ji}$ and equations 39 can be easily rearranged as follow

$$\begin{cases} \hat{x}_i(t+1) & = \hat{x}_i(t) + f_q(x_i(t) - \hat{x}_i(t), z_i(t)) \\ z_i(t+1) & = g_{k_1,k_2,q}(x_i(t) - \hat{x}_i(t), z_i(t)) \\ x_i(t+1) & = x_i(t) + \sum_{j:(i,j)\in\mathcal{E}} k_{ij}\hat{x}_j(t+1) \end{cases},$$

where $f_q$ and $g_{k_1,k_2,q}$ are two nonlinear functions defined by

$$\begin{aligned} f_q(x-\hat{x}, z) = &\quad q|z|\left(\tfrac{1}{2} + \left\lfloor \tfrac{x-\hat{x}}{q|z|} \right\rfloor\right), \\ g_{k_1,k_2,q}(x-\hat{x}, z) = &\quad k_1|z|\mathrm{sgn}\left(f_q(x-\hat{x}, z)\right) + \\ &\quad + k_2 f_q(x-\hat{x}, z). \end{aligned}$$

From the visualization of function $f_q(x_i - \hat{x}_i, z_i)$ presented in Fig. 19, it is once again clear that the smaller the zoom factor $z_i$ is, the more precise the difference between $x_i$ and $\hat{x}_i$ sent from transmitter to receiver would be, thus improving the estimation accuracy.

More compactly the equations of the system can be written as

$$\begin{cases} x(t+1) & = & x(t) + K\left[\hat{x}(t) + \Phi_q\left(x(t) - \hat{x}(t), z(t)\right)\right] \\ \hat{x}(t+1) & = & \hat{x}(t) + \Phi_q\left(x(t) - \hat{x}(t), z(t)\right) \\ z(t+1) & = & \Gamma_{k_1,k_2,q}\left(x(t) - \hat{x}(t), z(t)\right) \end{cases},$$

where we have grouped again the state variables into state vectors and we have collected the scalar nonlinear functions into two multidimensional functions

$$\Phi_q(x-\hat{x}, z) = \begin{bmatrix} f_q(x_1 - \hat{x}_1, z_1) \\ \vdots \\ f_q(x_N - \hat{x}_N, z_N) \end{bmatrix},$$

$$\Gamma_{k_1,k_2,q}(x-\hat{x}, z) = \begin{bmatrix} g_{k_1,k_2,q}(x_1 - \hat{x}_1, z_1) \\ \vdots \\ g_{k_1,k_2,q}(x_N - \hat{x}_N, z_N) \end{bmatrix}.$$

Finally, if we introduce the functions

$$\begin{cases} \overline{\Phi}_q(x-\hat{x}, z) = \Phi_q(x-\hat{x}, z) - (x-\hat{x}) \\ \overline{\Gamma}_{k_1,k_2,q}(x-\hat{x}, z) = \Gamma_{k_1,k_2,q}(\delta, z) - k_2(x-\hat{x}). \end{cases}, \tag{40}$$

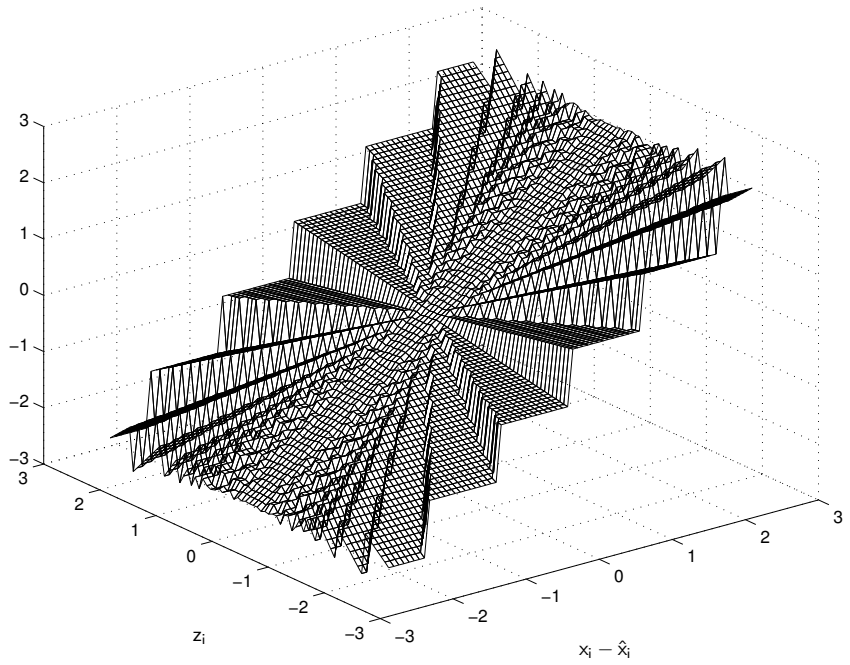


**Figure 19. A 3D visualization of the function** $f_{0.4}(x_i - \hat{x}_i, z_i)$

the system becomes

$$\begin{cases} x(t+1) = Px(t) + K\overline{\Phi}_q\left(x(t) - \hat{x}(t), z(t)\right) \\ \hat{x}(t+1) = x(t) + \overline{\Phi}_q\left(x(t) - \hat{x}(t), z(t)\right) \\ z(t+1) = k_2(x(t) - \hat{x}(t)) + \overline{\Gamma}_{k_1,k_2,q}\left(x(t) - \hat{x}(t), z(t)\right) \end{cases}. \tag{41}$$

We have to show now that, if the matrix $P$ is such that the linear algorithm 34 can achieve consensus, the system 41 is able to drive the agents' states to the consensus as well, more precisely

$$\lim_{t\to\infty} \begin{bmatrix} x(t) \\ \hat{x}(t) \\ z(t) \end{bmatrix} = \frac{\mathbf{1^T x_0}}{N} \begin{bmatrix} \mathbf{1} \\ \mathbf{1} \\ 0 \end{bmatrix} \quad \forall \begin{bmatrix} x_0 \\ \hat{x}_0 \\ z_0 \end{bmatrix} \in \mathbb{R}^{3N}. \tag{42}$$

To do this we need first to transform the consensus problem into a stability problem and then apply the small-gain theorem.

We start pointing out that, since $P$ is doubly stochastic, the system 41 is invariant in respect to any change of variables given by

$$y = x - \gamma\mathbf{1} \qquad \hat{\mathbf{y}} = \hat{\mathbf{x}} - \gamma\mathbf{1},$$

and, therefore, it suffices to prove 42 only for inital state with zero mean, that is

$$\lim_{t\to\infty} \begin{bmatrix} x(t) \\ \hat{x}(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \forall \begin{bmatrix} x_0 \\ \hat{x}_0 \\ z_0 \end{bmatrix} \in \mathbb{R}^{3N} : \mathbf{1^T x_0} = \mathbf{0}. \tag{43}$$

Moreover, from the first of 41, we obtain[4]

$$\mathbf{1^T x(t+1) = 1^T x(t)},$$

[4]We remark that $\mathbf{1^T P = 1^T}$ and $\mathbf{1^T K = 0^T}$.

which easily proofs that the mean of agents' states is preserved during algorithm execution or, in other words, that the system's trajectory will always lie in the subspace $\mathcal{U} = \left(\left[\mathbf{1^T}\ \mathbf{0^T}\ \mathbf{0^T}\right]^T\right)^{\perp}$ whenever the initial state has zero mean.

In order to obtain the equation of the system restricted to the subspace $\mathcal{U}$, we introduce the following linear transformation

$$\tilde{x} = Tx = \begin{bmatrix} I_{N-1} & 0 \\ \mathbf{1^T} & 1 \end{bmatrix} x = \begin{bmatrix} x_1 \\ \vdots \\ x_{N-1} \\ \mathbf{1^T x} \end{bmatrix} = \begin{bmatrix} \eta \\ \sigma \end{bmatrix}, \tag{44}$$

which is invertible since we have[5]

$$T^{-1} = \begin{bmatrix} I_{N-1} & 0 \\ -\mathbf{1^T} & 1 \end{bmatrix} = \left[\begin{array}{c|c} H & v \end{array}\right].$$

Using the relation $x = T^{-1}\tilde{x}$, the equations of the system in the new state space become

$$\begin{cases} \tilde{x}(t+1) = TPT^{-1}\tilde{x}(t) + TK\overline{\Phi}_q\left(T^{-1}\tilde{x}(t) - \hat{x}(t), z(t)\right) \\ \hat{x}(t+1) = T^{-1}\tilde{x}(t) + \overline{\Phi}_q\left(T^{-1}\tilde{x}(t) - \hat{x}(t), z(t)\right) \\ z(t+1) = k_2 T^{-1}\tilde{x}(t) - k_2\hat{x}(t) + \\ \qquad\qquad + \overline{\Gamma}_{q,k_1,k_2}\left(T^{-1}\tilde{x}(t) - \hat{x}(t), z(t)\right) \end{cases},$$

where the matrices $TPT^{-1}$ and $TK$ have many notable symmetries that can be appreciated introducing the following congruent partition on the matrices $P$ and $K$

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}, \quad P_{11} \in \mathbb{R}^{N-1\times N-1},$$

$$K = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}, \quad K_{11} \in \mathbb{R}^{N-1\times N-1}.$$

With this partition we obtain

$$\begin{aligned} TPT^{-1} &= \begin{bmatrix} I_{N-1} & 0 \\ \mathbf{1^T} & 1 \end{bmatrix}\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}\begin{bmatrix} I_{N-1} & 0 \\ -\mathbf{1^T} & 1 \end{bmatrix} \\ &= \begin{bmatrix} P_{11} - P_{12}\mathbf{1^T} & P_{12} \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} F & P_{12} \\ 0^T & 1 \end{bmatrix}, \\ TK &= \begin{bmatrix} I_{N-1} & 0 \\ \mathbf{1^T} & 1 \end{bmatrix}\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \\ &= \begin{bmatrix} K_{11} & K_{12} \\ 0^T & 0 \end{bmatrix} = \begin{bmatrix} G \\ 0^T \end{bmatrix}, \end{aligned} \tag{45}$$

[5] It is easy to see that $TT^{-1} = T^{-1}T = I_N$.

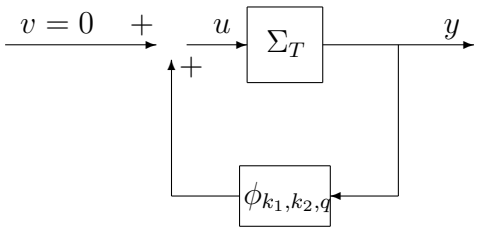


**Figure 20. Positive feedback connection between $\Sigma_T$ and $\phi_{k_1,k_2,q}$**

and rewriting system's equations with the partitioned state inducted by 44, reminding that if the initial state has zero mean then $\sigma(t) = \mathbf{1^T x(t)} = \mathbf{0} \quad \forall \mathbf{t} \geq \mathbf{0}$, the evolution of remaining variables is given by

$$\begin{cases} \eta(t+1) &= F\eta(t) + G\overline{\Phi}_q\left(H\eta(t) - \hat{x}(t), z(t)\right) \\ \hat{x}(t+1) &= H\eta(t) + \overline{\Phi}_q\left(H\eta(t) - \hat{x}(t), z(t)\right) \\ z(t+1) &= k_2 H\eta x(t) - k_2\hat{x}(t) + \\ & \quad +\overline{\Gamma}_{q,k_1,k_2}\left(H\eta(t) - \hat{x}(t), z(t)\right) \end{cases}, \tag{46}$$

where the state space has a smaller dimension of $3N - 1$.

It suffices to observe that the state variable in 46 are now unconstrained and that the property 43 requires $x$, and thus $\tilde{x}$ and $\eta$, to converge toward zero, to conclude that the convergence of the algorithm is ensured whenever the system 46 is proven to be globally asymptotically stable.

**Remark 3.** *Note that, since $P$ has an eigenvalue in $1$ while the others lie inside the unit circle, from 45 the matrix $F$ turns out to be asymptotically stable.*

To prove global asymptotic stability of system 46, we will use the small–gain theorem whose discrete–time version is well–presented in [31] (see also [44] for a dissertation on the bounded real lemma).

We start pointing out that the system 46 can easily be seen as a positive feedback interconnection between a linear system $\Sigma_T = (A_{k_2}, B, C)$ and a static nonlinear function $\phi_{k_1,k_2,q}$ as shown in Fig. 20.

The linear system's matrices are given by

$$A_{k_2} = \begin{bmatrix} F & 0 & 0 \\ H & 0 & 0 \\ k_2 H & -k_2 I_N & 0 \end{bmatrix} \in \mathbb{R}^{3N-1 \times 3N-1}$$

$$B = \begin{bmatrix} G & 0 \\ I_N & 0 \\ 0 & I_N \end{bmatrix} \in \mathbb{R}^{3N-1 \times 2N}$$

$$C = \begin{bmatrix} H & -I_N & 0 \\ 0 & 0 & I_N \end{bmatrix} \in \mathbb{R}^{2N \times 3N-1},$$

while the nonlinear function is defined as

$$\begin{array}{rccc} \phi_{k_1,k_2,q}: & \mathbb{R}^{2N} & \to & \mathbb{R}^{2N} \\ & y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} & \mapsto & \begin{bmatrix} \overline{\Phi}_q\left(y_1, y_2\right) \\ \overline{\Gamma}_{k_1,k_2,q}\left(y_1, y_2\right) \end{bmatrix}. \end{array}$$

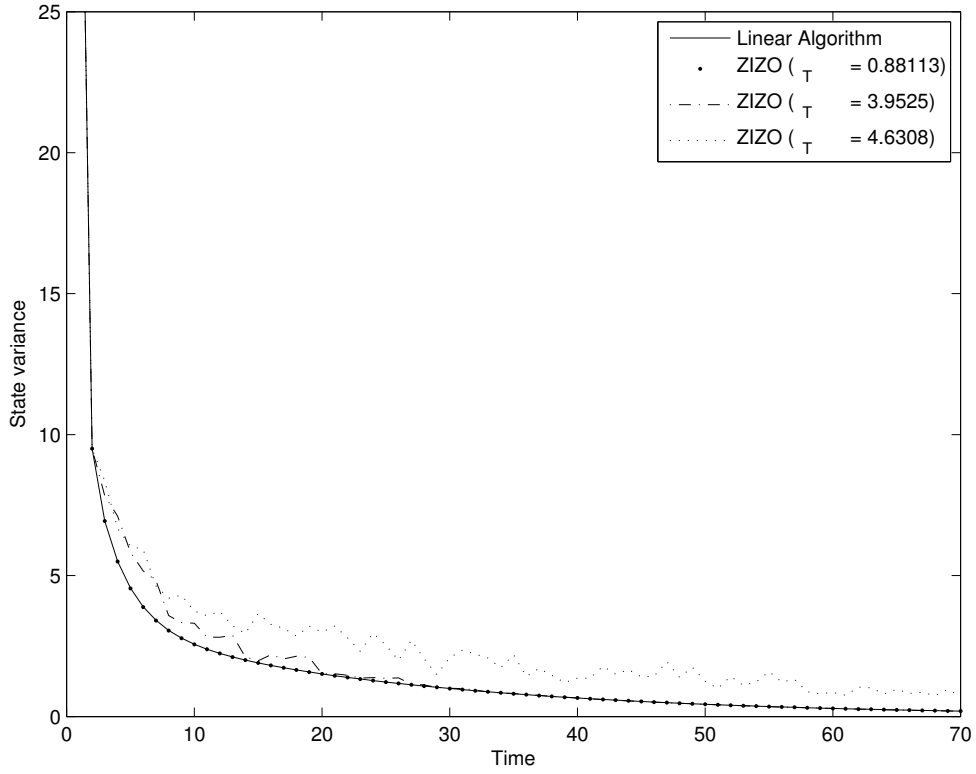


**Figure 21. State variance evolution with different parameters' choices**

We are now in position to apply the small–gain theorem. Let

$$\begin{aligned}\rho &= \max_{\substack{\vartheta\in[0,\pi]\\ \lambda\in\Lambda(P)}} \mu_\lambda(\vartheta) = \max_{\lambda\in\Lambda(P)} \mu_\lambda(\pi) \\ &= \max_{\lambda\in\Lambda(P)} \left(\tfrac{3-\lambda}{1+\lambda}\right)^2 = \left(\tfrac{3-\lambda_{\min}}{1+\lambda_{\min}}\right)^2,\end{aligned} \tag{47}$$

where $\lambda_{\min}$ is the minimum eigenvalue of $P$ and the last equality holds since $\left(\frac{3-\lambda}{1+\lambda}\right)^2$ is monotone decreasing in $(-1,\, 1]$. Then we have the following theorem.

**Theorem 7** (Convergence Theorem)**.** *Let $\mathcal{G} = (V, \mathcal{E})$ be a graph and let $P$ be an irreducible doubly stochastic symmetric matrix complying with the communication constraints given by $\mathcal{G}$. Let $\rho$ defined in 47. Then the ZIZO-Algorithm converges to the average-consensus regardless*[6] *of the initial choice of vectors $x_0$, $\hat{x}_0$ and $z_0$ if the three positive parameters of the algorithm $q$, $k_1$ and $k_2$ satisfy the following inequality:*

$$\left[\left(\tfrac{q}{2}\right)^2 + \left(k_1 + k_2\tfrac{q}{2}\right)^2\right] \times \\ \times \frac{1+(1+k_2^2)\rho+\sqrt{(k_2^2+1)^2\rho^2+2(k_2^2-1)\rho+1}}{2} < 1.$$

As a consequence of theorem 7, ZIZO–algorithm's parameters depend only on the minimum eigenvalue of the matrix $P$ and the convergence is assured whenever the matrix $P$ guarantee convergence

[6] As already pointed out, we need to force $z_0^i \neq 0$ to ensure the implementability of the algorithm. What really matter is the arbitrariness on the choice of $x_0$.

using the linear algorithm[7]. Since the minimum eigenvalue is usually independed of the number of agents[8] or can be forced to be greater to any constant $\beta$ without any further constraint on $\mathcal{G}$ providing that the distributed constraints $p_{ii} \geq \frac{\beta+1}{2}$ hold, it turns out that the algorithm's parameters can be chosen regardless to the dimension of $\mathcal{G}$.

In Fig. 21 we show some simulation results obtained with a random geometric graph with 20 agents. The weights' matrix $P$ was generated using Metropolis–algorithm, thus obtaining $\rho \simeq 12.7$, and the algorithm's parameters are chosen in order to obtain different values for the closed loop gain $\gamma_T\gamma_\phi$. From these numerical results it seems that the result stated in theorem 7 is still improvable since the algorithm seems to converge even for closed loop gain greater than 1; moreover the ZIZO–algorithm's performances seems very close to those of the linear algorithm 34 whenever the parameters fulfill the requirements of theorem 7 despite any bound on the convergence rate derivable from the small gain theorem seems very poor. This discrepancy between theoretical and sperimental results is not surprising and is due to the conservativeness of the small-gain theorem.

### 4.2 A coding method for consensus in case of noisy digital channels

While most of the literature on consensus algorithms has modeled communication constraints in the average consensus algorithm by a communication graph in which a link between two nodes is assumed to support the noise-free transmission of a real value, there is a clear demand for more realistic communication models. In fact, some recent work has addressed the cases of quantized communication, [36, 2, 3, 33, 26], packet losses [23], or transmission affected by additive noise [32, 40]. However, at our knowledge, there is no contribution yet toward the design of consensus algorithms on networks in which the communication links are modeled as digital noisy channels. For such networks, information-theoretic bounds on the performance of distributed computation algorithms have been established in [1, 18].

In the present document, we shall present and analyze distributed algorithms for average consensus on networks with digital noisy communication channels. The algorithms we propose combine the classical iterative linear consensus algorithm with coding schemes for the reliable transmission of real numbers on noisy channels, recently proposed in [19]. Our algorithm are fully distributed, in the sense that they do not require the agents to have any knowledge of the network structure. The main results consist in showing that such algorithms drive the agents arbitrarily close to average consensus, at the price of using a number of channel transmissions and in-node computations at most poly-logarithmic in the desired precision. Scaling properties as the number of agents grows are investigated as well, and the computation time is related to spectral properties of the underlying communication graph.

The poly-logarithmic growth in the desired precision of both communication and computational complexity of the consensus algorithms for networks with digital noisy channels has to be compared with the logarithmic growth characterizing many of the algorithms proposed in the literature for quantized transmission channels. As it will be argued, such a performance gap has mainly to be attributed to the different availability of feedback information in the two problems. Indeed, deterministic channels inherently include perfect noiseless feedback, as the transmitter knows exactly what the receiver is going to observe: such a feedback information is critical in the design of consensus algorithms for networks with quantized communication links, as the agents use it to coordinate among themselves. In

[7] This implies that $\mathcal{G}$ must be connected

[8] So it is in the wide family of random geometric graphs.

contrast, when noiseless channel feedback is not available –as in the problem addressed in this chapter–, coordination of the different agents becomes a much more challenging task.

General results on coding for interacting communication [43, 41] may suggest logarithmic times to possibly be achievable by embedding an efficient quantized consensus algorithm in a global error correcting coding scheme. However, as it has also been argued in [29], the tree-code constructions proposed in [43, 41] suffer from high computational complexity which likely prevents their practical implementation. Moreover, their global design requires each agent to have knowledge of the whole network topology, an assumption which contrasts the reconfigurability requirements. In contrast, the algorithms we shall propose do not require the agents to have any knowledge of the network topology, and their computational complexity can also be kept tractable.

### 4.2.1 Problem statement and proposed algorithm

#### Problem statement

We shall consider a set $\mathcal{V}$ of $N$ agents, possibly representing sensors in a wireless network, each having access to some partial information consisting in the observation of a scalar value, which for instance may represent the measurement of some physical quantity. The observation of agent $v$ will be denoted by $\theta_v \in [0,1]$, while $\theta = (\theta_{\mathbf{v}})$ will indicate the full vector of observations. The goal is for the network to compute the arithmetic average of such values, $\theta_{\text{ave}} = \frac{1}{N}\sum_v \theta_v$ by exchanging information among themselves and without a centralized computing or scheduling system.

Communication among the agents takes place as follows. At each time instant $t = 1, 2, \ldots$, every agent $v$ broadcasts a bit $a_v(t) \in \{0,1\}$ to a subset of agents, to be denoted by $\mathcal{N}_v^+ \subset V$. Every agent $w \in \mathcal{N}_v^+$ receives a possibly erased version $b_{v,w}(t) \in \{0,1,?\}$ of $a_v(t)$. The set of in-neighbors of agent $w$ will be denoted by $\mathcal{N}_w^- = \{v : w \in \mathcal{N}_v^+\}$, whereas $b_w(t) = (b_{v,w}(t))_{v\in\mathcal{N}_w^-}$ will denote the vector of signals received by agent $w$ at time $t$. We shall assume that, conditioned on the broadcasted bits $(a_v(t))_v$, the received signals $b_{v,w}(t)$ are mutually independent, with $b_{v,w}(t) = a_v(t)$ with probability $1-\varepsilon$, and $b_{v,w}(t) =?$ with probability $\varepsilon$, where $\varepsilon > 0$ is some erasure probability which -for the sake of simplicity- is assumed to remain constant in $t$, $v$ and $w$. Furthermore, erasures will be assumed independent in time. Distributedness of the computation/communication algorithm is then enforced by requiring that the bit $a_v(t)$ to be a function of the local information available to agent $v$ at time $t$, i.e. of its initial observation $\theta_v$, as well as the signals $\{b_v(s)\}_{1\le s<t}$ received by agent $v$ up to time $t-1$.

The communication setting outlined above can be conveniently described by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (the communication graph), whose vertices are the agents, and such that an ordered pair $(i,j)$ with $i \neq j$ belongs to $\mathcal{E}$ if and only if $j \in \mathcal{N}_i^+$ (equivalently $i \in \mathcal{N}_j^-$), i.e. if $i$ transmits to $j$ with erasure probability $\varepsilon < 1$. In all our results, we shall assume that the graph $\mathcal{G}$ is strongly connected, i.e. that any two nodes $u, v$ are connected by a directed path. We will also assume that $\mathcal{G}$ has self-loops on each vertex; this represents the fact that each node has access to its own information, which is equivalent to assume a noiseless channel available from $i$ to itself. The presence of self-loops ensures that $\mathcal{G}$ is aperiodic, i.e. the greatest common divisor of its cycle lengths is 1.

#### Proposed algorithm

Our idea is to use a traditional linear average-consensus algorithm, combined with a technique for transmission of real numbers over noisy digital broadcast channels, a joint source and channel coding

proposed in [19].

Consider the above-described scenario, specified by a set of $N$ vertices $\mathcal{V}$, a strongly connected communication graph $\mathcal{G}$, and an erasure probability $\varepsilon$. The ingredients of our algorithm are:

- a consensus matrix $P$, i.e., a doubly-stochastic primitive matrix adapted to $\mathcal{G}$, with non-zero diagonal entries;
- an increasing sequence of positive integers $\{\tau(k)\}_{k\in\mathbb{N}}$, such that $\lim_{k\to\infty}\tau(k)=+\infty$; $\tau(k)$ represents the number of bits that each node transmits at $k$-th iteration of the algorithm;
- a sequence of encoders, i.e., maps $\phi^{(k)}:[0,1]\to\{0,1\}^{\tau(k)}$;
- a sequence of decoders, i.e., maps $\psi^{(k)}:\{0,1,?\}^{\tau(k)}\to[0,1]$;

The usual linear average-consensus algorithm is:

$$\boldsymbol{x}(0)=\boldsymbol{\theta}\,,\qquad \boldsymbol{x}(k+1)=P\boldsymbol{x}(k) \tag{48}$$

i.e., at $k$-th iteration, node $i$ receives from its in-neighbors the numbers $x_j(k)$, $j\in\mathcal{N}_i^-$, and updates its state: $x_i(k+1)=\sum_{j\in\mathcal{N}_i^-}P_{ij}x_j(k)$.

We propose to adapt this algorithm, in a way that takes into account the necessity to transmit the real values $x_j(k)$ along digital noisy channels. The initialization of the algorithm is unchanged: $\boldsymbol{x}(0)=\boldsymbol{\theta}$. Between iterations $k$ and $k+1$ of our consensus-like algorithm, we allow each node $j$ to broadcast $\tau(k)$ bits to its neighbors:

- the bits transmitted by node $j$ at iteration $k$ are the message $a_j(k)=\phi^{(k)}(x_j(k))$, i.e., a suitably encoded version of the state $x_j(k)$;
- each $i\in\mathcal{N}_j^+$ will receive a corrupted version of $a_j(k)$, $b_{ij}(k)$, and will use the decoder $\psi^{(k)}$ to recover an estimate $\hat{x}_{ij}(k)=\psi^{(k)}(b_{ij}(k))$

Then, the next consensus iteration will take place, where node $i$ will use $\hat{x}_{ij}(k)$ to replace the exact state $x_j(k)$ which he can not know exactly:

$$x_i(k+1)=P_{ii}x_i(k)+\sum_{j\in\mathcal{N}_i^-}P_{ij}\hat{x}_{ij}(k)$$

Clearly, we can write $\hat{x}_{ij}(k)=x_j(k)+v_{ij}(k)$, and we might think at $v_{ij}(k)$ as a residual noise which could not be removed by the error-correction performed by the decoder. Notice that $v_{ij}(k)$ in general does not have zero-mean, and depends on $x_j(t)$ (and thus depends on all past noises). A suitable choice of the encoder/decoder pairs and of the transmission phases allows to obtain a noise deacreasing with respect to $k$, with a speed which will be discussed in Section 4.2.2. To this effect, the assumption that the transmission lengths $\tau(k)$ are increasing in $k$ is essential, because the coding gain is asymptotic in the length of codewords. This remarks leads us to name our proposed algorithm 'Increasing Precision Algorithm' (IPA).

We can summarize the evolution of the state $\boldsymbol{x}(k)$ in the IPA algorithm in the following compact form: define $V(k)$ to be the matrix with entries $V(k)_{ij}=v_{ij}(k)$ if $j\in\mathcal{N}_i^-$, and 0 otherwise; then:

$$\boldsymbol{x}(k+1)=P\boldsymbol{x}(k)+(V(k)\odot P(k))\,\mathbf{1}\,, \tag{49}$$

where $\mathbf{1}$ denotes a vector with entries all equal to 1 and $\odot$ denotes the entry-wise (Hadamard) product between two matrices.

Notice that $k$ takes into account the number of consensus-like updates. However, it is more reasonable to measure the elapsed time in terms of the number of transmissions: state $\boldsymbol{x}(k)$ is reached after $t(k) = \sum_{h \leq k} \tau(h)$ transmissions.

Another way of analyzing our algorithm is to take into account also computation time. In fact, the decoding process can be computationally relevant, in particular for codes with best error-correction performance. For this reason we introduce the two following definitions: the computation time per channel use $\kappa(\tau)$, which depends on the choice of the encoder/decoder pair and on the length $\tau$ used for transmission, and the total transmission/computation time

$$T(k) = \sum_{h \leq k} \max(\tau(h), \kappa(\tau(h))).$$

The choice of transmission time $t$ or total transmission/computation time $T$ as the relevant notion of time step depends if the focus is on the number of channel uses (which are power-expensive), or the actual time (particularly relevant in the asymptotic regime).

### 4.2.2 Performance analysis

#### Preliminaries

Before starting our analysis, we shortly recall some well-known definitions and results about the consensus matrix $P$ (from Perron-Frobenius theory), which we will use in our proofs.

A square matrix $P$ is *stochastic* if $P_{ij} \geq 0$ for all $i$ and $j$ and $P\mathbf{1} = \mathbf{1}$; a stochastic matrix such that $\mathbf{1}^T P = \mathbf{1}^T$ is called *doubly stochastic*. Given a nonnegative $N \times N$ matrix $P$, we define the induced graph $\mathcal{G}_M$ by taking a set $\mathcal{V}$ of $N$ vertices, and putting an edge $(j, i)$ in $E$ if and only if $M_{ij} > 0$. Given a graph $\mathcal{G}$ on $\mathcal{V}$, the matrix $P$ is *adapted* to $\mathcal{G}$ if $\mathcal{G}_P \subset \mathcal{G}$.

A non-negative square matrix is $P$ is primitive if $\mathcal{G}_P$ is strongly connected and aperiodic. A primitive stochastic matrix $P$ is known to have the eigenvalue 1 with multiplicity 1 and eigenvector $\mathbf{1}$, and all other eigenvalues with modulus strictly smaller than 1. Moreover, if $P$ is doubly-stochastic, $P^t \boldsymbol{v}$ converges for $t \to \infty$ to the average of the entries of $\boldsymbol{v}$.

Consider a doubly-stochastic matrix $P$ with positive diagonal elements and with strongly connected associated graph $\mathcal{G}_P$. Under these assumptions, both $P$ and $P^T P$ are doubly-stochastic and primitive. Hence, $P$ has largest singular value equal to 1 and all other singular values strictly smaller than 1. Define $\rho(P)$ to be the second largest singular value of $P$, and notice that for all vectors $\boldsymbol{v} \perp \mathbf{1}$, $\|P\boldsymbol{v}\| \leq \rho(P)\|\boldsymbol{v}\|$, where $\|\cdot\|$ denotes Euclidean norm.

#### Average quadratic error

To analyze the performance of our algorithm, we will study the distance from the average of initial values $\theta_{\text{ave}}$, at $k$-th iteration of (49):

$$\boldsymbol{e}(k) = \boldsymbol{x}(k) - \frac{1}{N}\left(\mathbf{1}^T \boldsymbol{x}(0)\right)\mathbf{1}\,.$$

We have that $\boldsymbol{e}(k)$ can be decomposed as

$$\boldsymbol{e}(k) = \boldsymbol{z}(k) + \zeta(k)\mathbf{1}$$

where $\boldsymbol{z}(k) = \boldsymbol{x}(k) - \left(\frac{1}{N}\mathbf{1}^T\boldsymbol{x}(k)\right)\mathbf{1}$ represents the distance from consensus (distance from average of current states), whereas $\zeta(k) = \frac{1}{N}\mathbf{1}^T\left(\boldsymbol{x}(k) - \boldsymbol{x}(0)\right)$ accounts for the distance between the current average and the average of the initial conditions.

Our main result is the following theorem:

**Theorem 8.** *Consider the IPA algorithm, and let $\rho = \rho(P)$, and $\boldsymbol{z}(k)$ and $\zeta(k)$ be defined as above. Assume that the sequence of matrices $\{V(k)\}_{k\in\mathbb{N}}$ satisfies the property that for all $k \in \mathbb{N}$, $\mathbb{E}[V_{n,m}(k)^2] \leq \alpha^{2k}$ for some $0 < \alpha < \rho$. Then, for all $k \in \mathbb{N}$,*

$$\mathbb{E}[\zeta^2(k)] \leq \frac{\alpha^2}{(1-\alpha)^2}$$

*and*

$$\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(k)\|^2] \leq \rho^{2k}\frac{1}{\left(1-\frac{\alpha}{\rho}\right)^2}$$

□

**Remark 4.** *Theorem 8 allows to provide a bound also on $\frac{1}{N}\mathbb{E}[\|\boldsymbol{e}(k)\|^2]$. Indeed, under the assumptions of Theorem 8, we have that*

$$\begin{aligned}\frac{1}{N}\mathbb{E}[\|\boldsymbol{e}(k)\|^2] &= \frac{1}{N}\mathbb{E}[\|\zeta(k)\mathbf{1} + \boldsymbol{z}(k)\|^2] \\ &= \frac{1}{N}\mathbb{E}\left[\|\zeta(k)\mathbf{1}\|^2 + \|\boldsymbol{z}(k)\|^2\right] \\ &\leq \frac{\alpha^2}{(1-\alpha)^2} + \rho^{2k}\left(1-\frac{\alpha}{\rho}\right)^{-2}.\end{aligned}$$

Note that Theorem 8 implies that, as $k \to \infty$, mean square consensus is asymptotically reached. Moreover observe, that the mean squared distance between the asymptotic consensus point and the average of the initial conditions, is upper bounded by $\frac{\alpha^2}{(1-\alpha)^2}$; since $\alpha$ depends only on the coding transmission scheme, this bound is, remarkably, independent of either the size of the network or the consensus matrix $P$.

### Achievable noise decay

Here, we shortly describe two families of codes which allow (at the price of a different complexity) to obtain two levels of decay speed of the error after decoding. More details can be found in [19].

A first family, which we will call class-(a) encoder/decoder pairs, is based on linear random-tree codes. Such coding schemes are guaranteed to have error estimated by $\mathbb{E}[(x-\hat{x})^2] \leq C\beta^{2\tau}$, where $\tau$ is the transmission length, and $C > 0$, $\beta \in (0,1)$, are constants depending on the erasure probability $\varepsilon$ only. The encoding complexity of these schemes grows quadratically in $\tau$, while the decoding complexity scales like $\tau^3$.

A second family, which will be referred to as class-(b), has both encoding and decoding complexity scaling linearly in $\tau$, and error which can be estimated by $\mathbb{E}[(x-\hat{x})^2] \leq C\beta^{2\sqrt{\tau}}$, for some constants $C > 0$, $\beta \in (0,1)$.

Clearly, performance of the coding schemes is given with respect to the transmission length $\tau$. The speed of convergence of the error to zero with respect to $\tau$ suggests the correct choice of time phases $\tau(k)$ to use in the IPA algorithm, in order to achieve exponential decay of the error with respect to iteration number $k$:

(a) if the encoder belongs to class (a), choose $\tau(k) = Sk$ for $S > 0$;

(b) if the encoder belongs to class (b), choose $\tau(k) = S^2k^2$ for some $S > 0$.

With this choice, the assumptions of Theorem 8 are met with $\alpha = \beta^S$. Notice that $\alpha$ can be made arbitrarily small by increasing $S$, but at the same time large $S$ corresponds to longer transmission time; in Section 4.2.2 we will discuss suitable choices of the parameter $S$.

We define the algorithm IPA-(a) and IPA-(b) respectively to be the IPA algorithm with a sequence of encoder/decoder pairs chosen from class (a) or resp. (b), and with the corresponding choice of $\tau(k)$ defined above. Clearly IPA algorithm is more general, but we focus on this particular choice, in order to give concretely some implementable options, for which we will provide in the next sections a detailed performance analysis and simulation results. Moreover, given the choice of a coding scheme within our proposed classes, then among all choices of transmission lengths $\tau(k) = \Theta(k^\eta)$, $\eta > 0$, our choice of $\eta$ is best possible.

Recalling that the total transmission time is $t(k) = \sum_{h\le k} \tau(h)$, and that the total transmission/computation time is $T(k) = \sum_{h\le k} \max(\tau(h), \kappa(\tau(h)))$, we summarize here the scaling among these indexes for our algorithms, for further use in the next section.

**IPA-(a)** $\tau(k) = Sk$ implies that the total transmission time is $\frac{1}{2}Sk^2 \le t(k) \le Sk^2$. The computational complexity is $\kappa(\tau) = K\tau^3$, so that the total transmission/computation time is $T(k) = \Theta(k^4)$

**IPA-(b)** $\tau(k) = S^2k^2$ implies that the total transmission time is $t(k) = \Theta(k^3)$. The computational complexity is $\kappa(\tau) = K\tau$, so that the total transmission/computation time is $T(k) = \Theta(k^3)$ .

### Convergence times

First of all, we can re-write the result in Theorem 8 expressing the decay of the error with respect to transmission time $t$ and total transmission/computation time $T$. With a slight abuse of notation, we will write $\boldsymbol{z}(t)$ for $\boldsymbol{z}(t(k))$ and $\boldsymbol{z}(T)$ for $\boldsymbol{z}(T(k))$. The results are summarized in the following corollary.

**Corollary 3.** *(a) For the IPA-(a) algorithm there exists constants $C, \bar{C} > 0$ and $\gamma, \bar{\gamma} \in (0,1)$ such that:*

$$\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(t)\|^2] \le C\gamma^{\sqrt{t}}$$

*and*

$$\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(T)\|^2] \le \bar{C}\bar{\gamma}^{\sqrt[4]{T}}$$

*(b) For the IPA-(b) algorithm there exists constants $C, \bar{C} > 0$ and $\gamma, \bar{\gamma} \in (0,1)$ such that:*

$$\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(t)\|^2] \le C\gamma^{\sqrt[3]{t}}$$

*and*

$$\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(T)\|^2] \le \bar{C}\bar{\gamma}^{\sqrt[3]{T}}$$

□

Now we investigate how much time is necessary to achieve a specified tolerance on the distance from average consensus. A traditional index to evaluate the performance of the standard consensus algorithm in terms of its speed, is defined as follows. Given $\delta > 0$, we define the *$\delta$-convergence time* as

$$k_\delta = \inf\{k \in \mathbb{N} \mid \frac{1}{N}\|\boldsymbol{e}(h)\|^2 \leq \delta, \ \ \forall h \geq k\}.$$

With this definition, for standard linear average consensus algorithm we have that, for $\delta$ small enough,

$$k_\delta \leq \frac{\log \delta^{-1}}{\log \rho^{-1}}. \tag{50}$$

To understand the performance of the proposed digital average consensus algorithm, we need to adapt the above definition. Just considering the number of algorithm steps, irrespective of the number of channel accesses which are used at each step, would not be appropriate. We rather want to consider as performance index the *$\delta$-transmission time $t_\delta$*, defined as

$$t_\delta = \inf\{t(k) \in \mathbb{N} \mid \frac{1}{N}\mathbb{E}[\|\boldsymbol{e}(h)\|^2] \leq \delta, \ \ \forall h \geq k\}.$$

If additionally one wants to keep into account the time required by computations, a suitable convergence index is the *$\delta$-computation/transmission time $T_\delta$*, defined as

$$T_\delta = \inf\{T(k) \in \mathbb{N} \mid \frac{1}{N}\mathbb{E}[\|\boldsymbol{e}(h)\|^2] \leq \delta, \ \ \forall h \geq k\}.$$

These two indexes are the object of the next two results.

**Corollary 4.** *Consider algorithm IPA and assume that the assumptions of Theorem 8 are met with $\alpha = \beta^S$. Then, $S$ can be chosen in order to ensure that, for $\delta$ small enough,*

- *for algorithm IPA-(a),*

$$t_\delta \leq \frac{1}{8 \log \beta^{-1}} \frac{\log^3 \delta^{-1}}{\log^2 \rho^{-1}}; \tag{51}$$

- *for algorithm IPA -(b)*

$$t_\delta \leq \frac{1}{32 \log^2 \beta^{-1}} \frac{\log^5 \delta^{-1}}{\log^3 \rho^{-1}}. \tag{52}$$

□

Reasoning similarly to the previous derivation, we can also argue the following result, regarding the $\delta$-computation/ transmission time.

**Corollary 5.** *Consider algorithm IPA and assume that the assumptions of Theorem 8 are met with $\alpha = \beta^S$. Then, $S$ can be chosen in order to ensure that, for $\delta$ small enough,*

- *for algorithm IPA-(a),*

$$T_\delta \leq \frac{K}{128 \log^2 \beta^{-1}} \frac{\log^7 \delta^{-1}}{\log^4 \rho^{-1}};$$

- *for algorithm IPA-(b)*

$$T_\delta \leq \frac{K}{32 \log^2 \beta^{-1}} \frac{\log^5 \delta^{-1}}{\log^3 \rho^{-1}}.$$

□

**Scaling properties**

Known results of spectral graph theory from [17] and [13] tell us that, for some sequences of structured graphs of increasing size (e.g. square lattices), the essential spectral radius goes to 1. This implies that the convergence time (50) of the standard linear consensus algorithm diverges. For instance, square lattices have $\rho = 1 - \Theta(1/N)$, and then $k_\delta = O(N)$, when $N \to \infty$.

How does the convergence time scale with $N$ for the IPA algorithms? For brevity, we shall consider only the transmission time $t_\delta$ for the IPA-(a) algorithm: the other cases follow a similar discussion. Its scaling properties can be argued from (51) and (52). If for instance we consider square lattices, $t_\delta = O(N^2)$. Then its scaling is worse than that of the traditional linear consensus algorithm.

A different treatment is appropriate when the threshold $\delta$ is chosen to depend on $N$. For instance, when averaging is performed to estimate an unknown scalar from $N$ noisy measurements, one would likely require $\delta = \Theta(1/N)$. In this important special case we have that the algorithm scales, on square lattices, as $t_\delta = O(N^2 \log^3 N)$. Instead, if we consider hypercube networks, which have $\rho = 1 - \Theta(\log^{-1} N)$, then algorithm IPA-(a) scales as $t_\delta = O(\log^5 N)$.

### 4.2.3 Simulation results and comparison with decreasing gains strategy

For implementing our algorithm, we have chosen a very low-complexity strategy: we have considered a particularly simple instance of class-(b) coding scheme, which is a generalization of repetition codes. The encoder $\psi^{(k)} : [0,1] \to \{0,1\}^{\tau(k)}$ is constructed as follows. Given $x \in [0,1]$, denote its diadic expansion by $x = \sum_{i \geq 1} c_i 2^{-i}$, $c_i \in \{0,1\}$. Then $\phi^{(k)}(x) \in \{0,1\}^\tau$ consists in transmitting the bits $c_i$'s and some repetitions of them which are more frequent for most significant bits, as follows

$$\phi^{(k)}(x) = (c_1,\ c_1, c_2,\ c_1, c_2, c_3,\ c_1, c_2, c_3, c_4,\ \ldots) \ .$$

The decoder $\psi^{(k)}$ sees a version of such vector where some of the transmitted bits are erased, and constructs a decoded $\hat{x} = \sum_{i \geq 1} d_i 2^{-i}$ as follows. First, notice that all $c_i$'s with $i > \nu(k)$ were not transmitted at all, where $\nu(k)$ is such that $\nu(k)(\nu(k)+1)/2 = \tau(k)$; the decoder will put $d_i = 0$ for $i > \nu(k)$. For bits $c_i$, $i \leq \nu(k)$, the decoder will put correctly $d_i = c_i$ if at least one of the repeated occurrences of $c_i$ in the transmitted word has been received un-erased, and otherwise will let $d_i$ be 0 or 1 uniformly at random.

To form an instance of IPA-(b) algorithm with such coding scheme, we have chosen transmission lengths $\tau(k) = \frac{k(k+1)}{2} \sim \frac{k^2}{2}$ (in this case, $\nu(k) = k$). Theorem 8 and its corollaries apply and predict convergence to consensus.

With simulations, we want to compare our algorithm with a different strategy which was used in previous literature to compute approximate averages running a consensus algorithm in the presence of noisy communications. We will refer to such family of algorithms as to 'decreasing gain' algorithms, because the key idea is to have time-varying gains, which give decreasing weight to information coming from neighbors, so as to avoid accumulating an amount of error growing to infinity with time. Algorithms exploiting this idea were presented independently by various authors (see [32] and [40]). More precisely, the algorithm is the following. After initializing $\boldsymbol{x}(0) = \boldsymbol{\theta}$, iterate:

$$x_i(k+1) = \Big(1 - \mu(k)\Big) P_{ii} x_i(k) + \mu(k) \sum_{j \in \mathcal{N}_i^-} P_{ij} \tilde{x}_{ij}(k)$$

where $\tilde{x}_{ij}(k) = x_j(k) + w_{ij}(k)$ is the version of $x_j(k)$ received by $i$, affected by noise, while $\mu(k) \in (0,1)$ are chosen to satisfy

$$\sum_{k\geq 0} \mu(k) = \infty \quad \text{and} \quad \sum_{k\geq 0} \mu^2(k) < \infty\,.$$

Such algorithms were designed for channels were real numbers can be transmitted and are affected by an additive noise with zero-mean, and independent from past history as well as from other channels in the network. Under such assumptions, [32] and [40] use techniques of stochastic approximation theory to prove convergence to consensus.

However, we might apply them also to our digital noisy networks, if we replace $\tilde{x}_{ij}(k)$ with the value $\hat{x}_{ij}(k)$ obtained after the process of encoding – transmitting over the channel from $i$ to $j$ – decoding, by some suitable coding scheme. What we want to compare is the strategy of increasing transmission lengths versus that of decreasing gains, where we plug into the decreasing gain algorithm a coding/encoding of fixed length $\bar{\tau}$, not varying with $k$.

In the example of implementation that we propose, we choose the same simple repetition-like coding scheme described above, with a transmission length $\bar{\tau} = 15$ and we use gains $\mu(k) = \frac{1}{k}$.

Here we present one example from our simulations. We consider $N = 30$ agents, and a communication graph which is a strongly connected realization of a two-dimensional random geometric graph, where vertices are $30$ points uniformly distributed in the unit square, and there is a pair of edges $(i,j)$ and $(j,i)$ whenever points $i, j$ have a distance smaller than $0.4$. The erasure probability on the links is $\varepsilon = 0.5$. The initial condition $\boldsymbol{\theta}$ is randomly chosen inside $[0,1]^N$.

The communication graph is undirected, in the sense that $\mathcal{N}_i^- = \mathcal{N}_i^+$ for all $i \in \mathcal{V}$. So we choose to use, for both algorithms that we are comparing, a consensus matrix built according to the Metropolis weights rules for undirected graphs, illustrated in [48], which can be constructed distributedly, using only information on neighbors, as follows:

$$P_{ij} = \begin{cases} \frac{1}{1+\max\{\deg(i),\deg(j)\}} & \text{if } (i,j) \in \mathcal{E} \\ 1 - \sum_{k\in\mathcal{N}_i^-} P_{ik} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

where $\deg(i)$ is the number of neighbors of node $i$.

Figures 22 and 23 show the decay of $\frac{1}{N}\mathbb{E}[\|\boldsymbol{z}(t)\|^2]$ and $\frac{1}{N}\mathbb{E}[\|\boldsymbol{e}(t)\|^2]$ respectively, with respect to the number of transmissions $t$. All our simulations show a similar behavior, where our algorithm outperforms the decreasing gain strategy.

## References

[1] O. Ayaso, D. Shah, and M. Dahleh (2008). Information theoretic bounds for distributed computation. *IEEE Transactions on Information Theory*. Submitted.

[2] T. C. Aysal, M. Coates, and M. Rabbat (2007). Distributed average consensus using probabilistic quantization. In *IEEE Workshop on Statistical Signal Processing*, 640–644, Maddison, Wisconsin.

[3] T. C. Aysal, M. Coates, and M. Rabbat (2008). Distributed average consensus with dithered quantization. *IEEE Transactions on Signal Processing*, 56(10), 4905–4918.

[4] G. Baldan and S. Zampieri (2009). An efficient quantization algorithm for solving average-consensus problems. In *Proceedings of the European Control Conference*, Budapest.

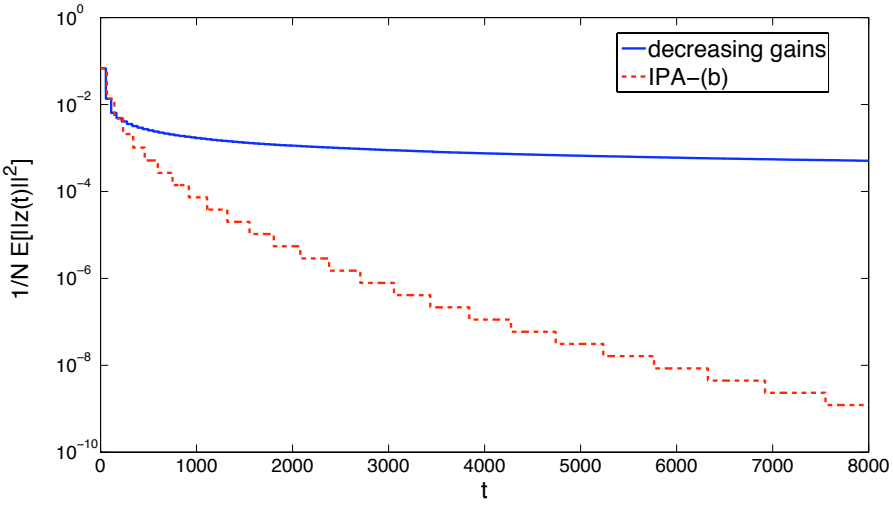


**Figure 22. Behavior of $\frac{1}{N}\mathbb{E}[\|z(t)\|^2]$ with the respect to the number of transmissions $t$.**

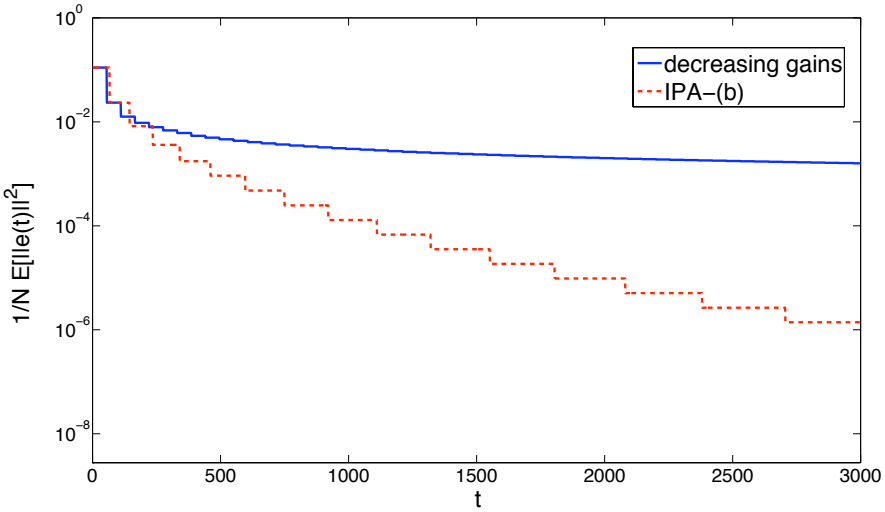


**Figure 23. Behavior of $\frac{1}{N}\mathbb{E}[\|e(t)\|^2]$ with the respect to the number of transmissions $t$.**

[5] P. Barooah and J. Hespanha (2005). Estimation from Relative Measurements: Error Bounds from Electrical Analogy In Proceedings of ICISIP.

[6] S. Boyd, P. Diaconis, P. Parrilo, and L. Xiao (2005). Symmetry analysis of reversible markov chains. *Internet Mathematics*, 2:31–71.

[7] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah (2006). Randomized gossip algorithms. *IEEE/ACM Trans. Netw.*, 14(SI):2508–2530.

[8] F. Bullo, J. Cortes, and S. Martinez (2008). *Distributed Control of Robotic Networks*. Available at http://coordinationbook.info.

[9] R. Carli, J. Delvenne and S. Zampieri (2009). Optimal strategies in the Average Consensus Problem. *Systems & Control Letters*, 58: 759–765.

[10] R. Carli, G. Como, P. Frasca, and F. Garin (2009). Average consensus on digital noisy networks. In *Proceedings of 1st IFAC Workshop on Estimation and Control of Networked Systems*, Venice, Italy.

[11] R. Carli, F. Fagnani, P. Frasca, T. Taylor, S. Zampieri (2007). Average consensus on networks with transmission noise or quantization. In *Proceedings of the European Control Conference*.

[12] R. Carli, F. Fagnani, P. Frasca, S. Zampieri (2007). Efficient quantized techniques for consensus algorithms. In *Proceedings of the 3rd international workshop on Networked Control Systems: Tollerant to faults*, Nancy, France.

[13] R. Carli, F. Fagnani, A. Speranzon, and S. Zampieri (2008). Communication constraints in the average consensus problem. *Automatica*, 44(3), 671–684.

[14] R. Carli, P. Frasca, F. Fagnani and S. Zampieri (2010). Gossip consensus algorithms via quantized communication. *Automatica*, 46: 70–80.

[15] R. Carli, F. Garin and S. Zampieri (2009). Quadratic indexes for the analysis of consensus algorithms. In *Proceedings of Information Theory and Applications Workshop*, San Diego CA USA, 96–104.

[16] P. Chen and S. Sastry (2007). Latency and connectivity analysis tools for wireless mesh networks. In *Proceedings of the First International Conference on Robot Communication and Coordination (ROBOCOMM)*.

[17] F. R. K. Chung, (1997). *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Conference Board of the Mathematical Sciences, Washington, DC.

[18] G. Como, and M. Dahleh (2009). Lower bounds on the estimation error in problems of distributed computation. In *Information Theory and Applications*. San Diego, CA.

[19] G. Como, F. Fagnani, and S. Zampieri (2008). Anytime reliable transmission of real-valued information through digital noisy channels. *SIAM Journal on Control and Optimization*. Submitted.

[20] N. G. de Bruijn (1946). A Combinatorial Problem. *Koninklijke Nederlandse Akademie v. Wetenschappen*, 49, 758–764.

[21] P.G. Doyle and J.L. Snell (1984). Random Walks and Electric Networks In Mathematical Association of America, Carus Monographs.

[22] Dust Networks, Inc. (2006). "Technical overview of time synchronized mesh protocol (TSMP)," http://www.dustnetworks.com/docs/TSMP_Whitepaper.pdf.

[23] F. Fagnani and S. Zampieri (2009). Average Consensus with Packet Drop Communication. *Siam Journal on Control and Optimization*, 48: 102–133.

[24] P. Fraigniaud and P. Gauron (2006). D2b: A de Bruijn based content-addressable network. *Theor. Comput. Sci.*, 355(1):65–79.

[25] M. Franceschetti and R. Meester (2007). *Random networks for communication*. Cambridge University Press, Cambridge.

[26] P. Frasca, R. Carli, F. Fagnani, and S. Zampieri (2009). Average consensus on networks with quantized communication. *International Journal of Robust and Nonlinear Control*, 19:1787–181.

[27] F. Garin and S. Zampieri (2009). Performance of consensus algorithms in large-scale distributed estimation. In *Proceedings of the European Control Conference*, Budapest.

[28] E. N. Gilbert (1961). Random plane networks. *Journal of SIAM*, 9:533–543.

[29] A. Giridhar, and P. R. Kumar (2006). Towards a theory of in-network computation in wireless sensor networks. *IEEE Communications Magazine*, 44(4):98–107.

[30] P. Gupta and P. Kumar (2000). The capacity of wireless networks. *Information Theory, IEEE Transactions on*, 46(2):388–404.

[31] W. M. Haddad, D. S. Bernstein (1994). Explicit construction of quadratic Lyapunov functions for the small gain, positivity, circle, and popov theorems and their application to robust stability. Part II: discrete–time theory. *International journal of robust and nonlinear control,* 4:249–265.

[32] M. Huang, and J. H. Manton (2009). Coordination and consensus of networked agents with noisy measurements: Stochastic algorithms and asymptotic behavior. *SIAM Journal on Control and Optimization*, 48(1), 134–161.

[33] S. Kar, and J. M. F. Moura (2007). Distributed consensus algorithms in sensor networks: Quantized data. Submitted.

[34] E. Lovisari and F. Garin and S. Zampieri (2010). A resistance-based approach to consensus algorithm performance analysis. In *Proceedings MTNS 2010*.

[35] E. Lovisari and F. Garin and S. Zampieri (2010). A resistance-based approach to performance analysis of the consensus algorithm. In *Proceedings 49th IEEE Conference on Decision and Control*.

[36] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis (2007). On distributed averaging algorithms and quantization effects. Technical Report 2778, LIDS-MIT.

[37] R. Olfati-Saber and R. M. Murray (2004). Consensus problems in networks of agents with switching topology and time-delays. *Automatic Control, IEEE Transactions on*, 49(9):1520–1533.

[38] M. Penrose (2003). *Random Geometric Graphs*. Oxford University Press.

[39] S. Rai (2007). The spectrum of a random geometric graph is concentrated. *J. Theoret. Probab.*, 20(2):119–132.

[40] R. Rajagopal, and M .J. Wainwright (2008). Network-based consensus averaging with general noisy channels. *CoRR*, abs/0805.0438. On-line available.

[41] S. Rajagopalan, and L. J. Schulman (1994). A coding theorem for distributed computation. In *Annual ACM Symposium on Theory of Computing*, 790–799. Montréal, Québec, Canada.

[42] L. Schenato (2006). Optimal estimation in networked control systems subject to random delay and packet loss. In *Proceedings of the 45th IEEE Conference on Decision and Control*.

[43] L. J. Schulman (1996). Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6), 1745–1756.

[44] C. E. de Souza, L. Xie (1992). On the discrete–time Bounded Real Lemma with application in the characterization of static state feedback $H_\infty$ controllers. *Systems & Control Letters* 18:61–71.

[45] D. Varagnolo, P. Chen, L. Schenato and S. Sastry (2008). Performance analysis of different routing protocols in wireless sensor networks for real-time estimation. In *Proceedings of the 47th IEEE Conference on Decision and Control*.

[46] L. Xiao and S. Boyd (2004). Fast linear iterations for distributed averaging. *Systems and Control letters*, 53(1):65–78.

[47] L. Xiao, S. Boyd, and S.-J. Kim (2007). Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46.

[48] L. Xiao, S. Boyd, and S. Lall (2005). A scheme for asynchronous distributed sensor fusion based on average consensus. In *International Conference on Information Processing in Sensor Networks (IPSN'05)*, pages 63–70, Los Angeles, CA.

[49] F.J. Zhang and G.N. Lin (1987). On the de Bruijn-Good graphs. *Acta Math. Sinica*, 30(2):195–205.